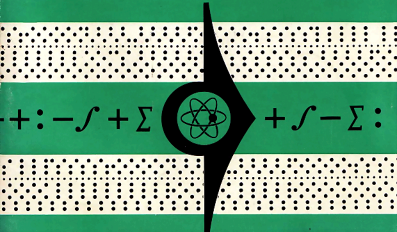
CELLATRON

VEB
KOMBINAT ZENTRONIK
RECHENELEKTRONIK
MEININGEN/ZELLA-MEHLIS

PROGRAMMIERUNG UND BEDIENUNG DES DIGITALEN KLEINRECHNERS C 8205

HEFT 2

ANLEITUNG ZUR PROGRAMMIERUNG IN MASCHINENSPRACHE



Nachdruck, Reproduktionen und Nachbildung – auch auszugsweise – nur mit Genehmigung des Herstellerbetriebes gestattet.

Vorwort

Das vorliegende Heft 2 der Anleitung zur Programmierung und Bedienung des digitalen Kleinrechenautomaten C 8205 behandelt die Maschinensprache des Rechners und ist in erster Linie als Unterlage für einen Programmierungslehrgang gedacht. Das erfolgreiche Studium der Maschinensprache setzt Kenntnisse im Umfang von Heft 1, Grundbegriffe der Programmierung, voraus.

Die praktische Programmierung erfolgt im wesentlichen nicht in Maschinensprache, sondern mittels Programmierungshilfen in Form von Interpretationssystemen, Assemblern und Compilern. Diese Programmierungshilfen sind in Maschinensprache programmiert.

Die Behandlung von Wortinterpretationen beschränkt sich auf typische Beispiele. Korrekturmöglichkeiten bei Tippfehlern werden dabei durch die Interpretationssysteme festgelegt.

Da für die vollständige Befehlsliste auch eines in Maschinensprache geschriebenen Programmes die Eingabeorganisation eines Interpretationssystems notwendig ist, tragen die Beispiele in Maschinensprache nur Übungscharakter.

Inhaltsverzeichnis

Vorwort

1. Allgemeine Automatenbeschreibung

2. Funktioneller Aufbau
2.1. Blockschaltbild
2.2. Befehlsschleife
2.2.1. Phase I
2.2.2. Phase II
2.2.3. Phase III
2.2.4. Phase IV
2.3. Speicherorganisation

3. Maschinenwort
3.1. Interne Darstellung
3.1.1. Dualzahlen
3.1.2. Befehle
3.2. Triadendarstellung

4. Befehlssystem
4.1. Befehlstyp
4.1.1. Organisationsbefehl 1. Art
4.1.2. Organisationsbefehl 2. Art
4.1.3. Einzelbefehl
4.1.4. Markierter Einzelbefehl
4.1.5. Gruppenbefehl
4.1.6. Verkürzter Gruppenbefehl
4.1.7. Wiederholungsbefehl
4.1.8. Verkürzter Wiederholungsbefehl
4.2. Operationszusatz
4.2.1. Resultatbehandlung
4.2.2. Löschbit
4.2.3. Zusammenfassung von Resultatbehandlung und Löschbit
4.3. Operationsart
4.3.1. Nichtorganisatorische Befehle
4.3.1.1. Konjunktion im Akkumulator
4.3.1.2. Addition
4.3.1.3. Subtraktion
4.3.1.4. Bedingte Addition (Subtraktion)
4.3.1.5. Reines Vergleichersetzen
4.3.1.6. Transport
4.3.1.7. Konjunktion im Speicher
4.3.1.8. Konjunktion mit negiertem Akkumulatorinhalt
4.3.1.9. Bedingte Speicherlöschung
4.3.1.10. R-Operation
4.3.2. Organisationsbefehle 1. Art
4.3.2.1. Unbedingter Sprung
4.3.2.2. Eingabe-Halt 1. Art
4.3.2.3. Unbedingter Stop 1. Art
4.3.2.4. Sprung mit Rückkehrabsicht
4.3.2.5. Ausgabe-Halt
4.3.2.6. Negativsprung
4.3.2.7. Positivsprung
4.3.2.8. Bedingter Stop 1. Art
4.3.2.9. Bedingter Sprung mit Rückkehrabsicht
4.3.2.10. Überlaufsprung
4.3.3. Organisationsbefehle 2. Art
4.3.3.1. Unbedingter Sprung
4.3.3.2. Eingabe-Halt 2. Art
4.3.3.3. Unbedingter Stop 2. Art
4.3.3.4. Sprung mit Rückkehrabsicht

4.3.3.5. Ausgabe-Halt
4.3.3.6. Vergleichersprung 1
4.3.3.7. Vergleichersprung 2
4.3.3.8. Bedingter Stop 2. Art
4.3.3.9. Leer-Befehl
4.3.3.10. Vergleichersprung 3
4.4. Liste der Operationen
4.5. Zusammenstellung der häufigsten Operationen

5. Beispiele
5.1. Vertauschung zweier Wörter
5.2. Algebraische Summe
5.3. Ausblendung der Adreßstellen
5.4. Multiplikation mit 10
5.5. Algebraischer Ausdruck
5.6. Vordere Bits als Adresse
5.7. Trennung des Maschinenwortes in Teilwörter
5.8. Betrag einer Summe
5.9. Ordnen eines Zahlenpaares
5.10. Addition und Zahlenbereichsüberschreitung
5.11. Angabe des Vorzeichens einer Zahl
5.12. Adresseneingabe
5.13. Dezimaleingabe
5.14. Abdruck numerischer Texte vom Lochband (im Telegrafenalphabet Nr. 2)
5.15. Abdruck numerischer Texte vom Lochband (im R 300-Code)
5.16. Umspeichern eines Vektors
5.17. Oktaldruck
5.18. Kleines Eingabeprogramm
5.19. Zeitoptimale Programmierung
5.19.1. Algebraischer Ausdruck
5.19.2. Trennung des Maschinenwortes in Teilwörter

6. Wortinterpretationen
6.1. Zahlwort „Festkomma vorn"
6.1.1. Multiplikation
6.1.2. Verkürzte Multiplikation
6.1.3. Quadrierung
6.1.4. Division
6.2. Ganze Festkommazahlen
6.2.1. Multiplikation
6.3. Gleitkommazahlen
6.3.1. Betragsbildung
6.3.2. Vorzeichenumkehr
6.3.3. Ordnen eines Zahlenpaares nach dem Betrag

Anhang 1: Telegrafenalphabet Nr. 2

Anhang 2: R 300-Lochbandcode

1. Allgemeine Automatenbeschreibung

Der Cellatron C 8205 ist ein programmgesteuerter digitaler Kleinrechenautomat, der in Serie arbeitet. Der Rechner ist volltransistorisiert; die Zentraleinheit enthält ca. 400 Transistoren und 2000 Dioden. Dieser relativ geringe Aufwand wird durch Beschränkung auf ein Rechenregister (Akkumulator), mehrfache Ausnutzung von Baugruppen sowie durch Verzicht auf die Verdrahtung von Multiplikation und Division erreicht. Um trotzdem eine brauchbare Operationsgeschwindigkeit zu erzielen, werden Befehlstypen, die besondere Eigenschaften des Trommelspeichers zum Rechnen ausnutzen, eingeführt. Bei zeitoptimaler Programmierung können ca. 2000 verdrahtete Operationen pro Sekunde ausgeführt werden. Der C 8205 ist eine Einadreß-Maschine. Als Speichereinheit ist eine Magnettrommel mit einer Kapazität von 4096 Worten zu 33 Bits vorhanden. Die Arbeitsfrequenz des Rechners beträgt ca. 316 kHz; das entspricht einer Trommelfrequenz von ca. 18 000 Umdrehungen pro Minute. ≈ 300 Umdr./sec.

$4096 = 2^{12}$
1 Bit ≙ 4Byte

Das weitgehend analytische Befehlssystem ist außerordentlich flexibel. Mit nur 19 Grundoperationen lassen sich alle Befehle aufbauen.

Der C 8205 setzt sich zusammen aus:

1. Zentraleinheit mit Anzeige und Tastatur
2. Peripherie:

   2 Lochbandleser (Lesegeschwindigkeit maximal 120 Zeichen pro Sekunde)

   1 Lochbandstanzer (Stanzgeschwindigkeit maximal 50 Zeichen pro Sekunde)

   1 alphanumerisch ansteuerbares Schreibwerk für Ein- und Ausgabe (Ausschreibgeschwindigkeit 10 Zeichen pro Sekunde)

Die gesamte Anlage besteht aus einer Tischkombination.

Eine vielseitige Verwendbarkeit des Lochbandes als Informationsträger ist durch die angegebene Peripherie gewährleistet. Für die Lochbandeingabe im R 300-Code bzw. im Telegrafenalphabet Nr. 2 gibt es zwei hard-ware Varianten des Rechners. Dank seiner peripheren Ausstattung ist der Rechner nicht nur für wissenschaftlich-technische, sondern auch für ökonomische Aufgaben der Kleindatenverarbeitung mit Erfolg einsetzbar.

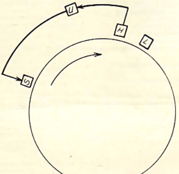


Abb. 1

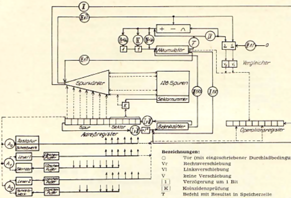


Abb. 2

Bezeichnungen:

○	Tor (mit eingeschriebener Durchlaßbedingung)
Vr	Rechtsverschiebung
Vl	Linksverschiebung
V	keine Verschiebung
[1]	Verzögerung um 1 Bit
[K]	Koinzidenzprüfung
T	Befehl mit Resultat in Speicherzelle
RS	Sprung mit Rückkehrabsicht
$\tilde{T}$	kein Befehl mit Resultat in Speicherzelle und kein Organisationsbefehl
F	Löschanweisung in Phase III
A_i	Anwahl des i-ten Kanals (i = 0, 1, 2)
V_i	Vergleicher i (i = 1, 2)
[+ − ∧]	Rechenwerk mit Verknüpfungen
[≥ ≤]	Vergleichersetzung
R	R-Befehl

2. Funktioneller Aufbau

2.1. Blockschaltbild

Bei der seriellen Arbeitsweise des C 8205 werden die einzelnen Bits generell in der Reihenfolge vom niedrigsten zum höchsten übertragen und verarbeitet.

Der C 8205 ist ein Einadreßrechner; demzufolge werden die Befehle in der natürlichen Reihenfolge der Speicherzellen aufgerufen und abgearbeitet. Die Steuerung dieser Reihenfolge erfolgt über den Befehlszähler, dessen 12-dualstelliger Inhalt die Adresse des nächsten Befehls im Sinne der natürlichen Reihenfolge der Befehlsabarbeitung ist und nach Abarbeitung eines Befehls um eine Adresseneinheit erhöht wird. Durch Sprungbefehle kann die natürliche Reihenfolge der Befehlsabarbeitung durchbrochen werden; der Befehlszähler wird neu eingestellt.

Befehlszähler und Akkumulator sind als Umlaufspeicher auf der Magnettrommel realisiert. Die prinzipielle Anordnung eines Umlaufspeichers ist durch Abbildung 1 gegeben. Die einzelnen Bits des Wortes werden über den Sprechkopf S nacheinander auf die Trommel aufgesprochen.

Dabei werden nur die Bits mit dem Wert 1 gesprochen, da durch den Löschkopf L alle Bits zu 0 gelöscht werden. Durch die Drehung der Trommel unter den räumlich feststehenden Köpfen wird die Information bis zum Hörkopf H mitgenommen. Dort wird sie abgehört, anschließend im Umformer U wieder zum Aufsprechen vorbereitet und, nachdem sie beim Sprechkopf S angelangt ist, wieder aufgesprochen. Das Wort hat also keinen festen Platz auf der Trommel, sondern läuft im beschriebenen Zyklus um. Wird dieser zwischen H und S unterbrochen, so wird durch den Löschkopf das Wort automatisch gelöscht.

Der 9-dualstellige Operationsteil des zur Abarbeitung aufgerufenen Befehls wird im Operationsregister, sein 12-dualstelliger Adreßteil im Adreßregister aufgestellt.

1° = 1 in Adreßeinheiten

BZ = Befehlszähler

AR = Adreßregister

Befehl: a ω (ω = BAZ)

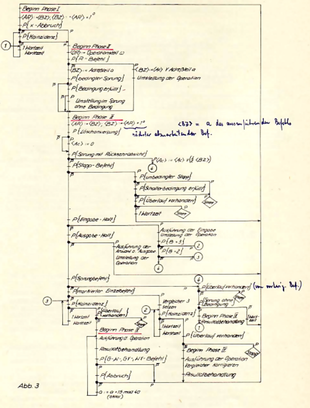


Abb. 3

Überlaufsprung kann auch programmiert werden. Dann wird FÜ nicht angezeigt.

Bei Zugriff zum Speicher erfolgt die Auswahl der Spur durch die Spurnummer im Adreßregister über eine Matrixschaltung, die Koinzidenzprüfung zur Auswahl des Sektors in der Spur erfolgt durch Vergleichen mit der Sektornummer im Adreßregister (s. auch 2.3.). Operationsregister und Adreßregister bilden zusammen das Befehlsregister. Das Befehlsregister ist als Schiebekette realisiert.

In einem Flip-Flop-Paar kann einer von drei definierten Vergleichszuständen bis zur Einstellung eines neuen Vergleichszustandes gespeichert werden.

Die Ein- und Ausgabe mittels der peripheren Geräte geschieht mit Ausnahme der manuellen Eingabe 1—Zeichen—gepuffert über das Adreßregister. Die Fortsetzung der Befehlsabarbeitung wird durch Umstellung der Operation im Operationsregister und deren Ausführung realisiert.

Eine Umstellung der Operation im Operationsregister wird auch bei bedingten Sprüngen mit erfüllter Bedingung vorgenommen. (Abb. 2)

2.2. Befehlsschleife

Der innere Ablauf bei der Abarbeitung von Maschinenbefehlen erfolgt in vier Phasen. Jede Phase dauert im allgemeinen eine Wortzeit (ca. 0,1 ms); Phase IV kann mehrere Wortzeiten benötigen. Gegebenenfalls können durch das Suchen nach der aufgerufenen Speicherzelle noch zusätzliche Wartezeiten, die dann natürliche Vielfache der Wortzeit sind, auftreten. Das Zusammenwirken der vom Programmierer zu beachtenden Teilabläufe innerhalb der Befehlsschleife ist in einem Ablaufplan vereinfacht dargestellt. (Abb. 3)

2.2.1. Phase I

Zu Beginn möge im Befehlszähler BZ die Adresse b des nächsten abzuarbeitenden Befehls stehen. Diese Adresse läuft in die 12 Dualstellen des Adreßregisters AR ein; geichzeitig wird der um eine Adreßeinheit erhöhte Inhalt des Adreßregisters in den Befehlszähler BZ transportiert.

$$\langle AR\rangle := \langle BZ\rangle$$
$$\langle BZ\rangle := \langle AR\rangle + 1^{\circ} .$$

Im Akkumulator Ac steht nach dieser Phase noch das Resultat R_{-1} der vorangegangenen Operation, welches für die nächste Operation benutzt werden kann, und im Operationsregister OR steht noch die alte Operationsanweisung ω_{-1}.

Der Informationszustand nach Phase I ist also durch $\langle AR\rangle := b$, $\langle OR\rangle = \omega_{-1}$, $\langle Ac\rangle = R_{-1}$ gegeben.

2.2.2. Phase II

Vor dem Einschalten der Phase II kann bis zur Koinzidenz (Übereinstimmung) zwischen der im Adreßregister aufgestellten Sektornummer und der Trommelstellung eine Wartezeit W_1 auftreten.

Bei Erreichen der Koinzidenz wird der Inhalt der Speicherzelle b gelesen und als Befehl interpretiert. Der 9-dualstellige Operationsteil ω wird dabei in das Operationsregister transportiert, der 12-dualstellige Adreßteil a wird im Befehlszähler zwischengespeichert:

$$\langle OR\rangle := \omega ,$$
$$\langle BZ\rangle := a .$$

Der Informationszustand am Ende der Phase II ist

$$\langle BZ\rangle = a , \langle AR\rangle = b , \langle OR\rangle = \omega , \langle Ac\rangle = R_{-1} .$$

Wenn der vorangegangene Befehl einen bestimmten Befehlstyp (GX- oder WX-Befehl) darstellt und seine Ausführung unter bestimmten Bedingungen endet, entfällt die Koinzidenzprüfung vor Phase II.

Die neue Operationsanweisung kann bereits in Phase II in den Befehlsablauf eingreifen. Bei den sogenannten R-Befehlen wird der Adreßteil im Befehlszähler disjunktiv mit den Adreßstellen des Akkumulatorinhaltes verknüpft und außerdem der

Operationsteil umgestellt. Bedingte Sprungbefehle werden bei erfüllter Bedingung in solche ohne Bedingung umgestellt. (Als bedingte Sprungbefehle gelten Negativsprung, Positivsprung, Überlaufsprung, Vergleichersprung 1, Vergleichersprung 2, Vergleichersprung 3 und bedingter Sprung mit Rückkehrabsicht).

2.2.3. Phase III

Unmittelbar an Phase II schließt sich Phase III an. Der im Befehlszähler zwischengespeicherte Adreßteil a läuft in das Adreßregister ein, während gleichzeitig der um eine Adresseneinheit erhöhte Adreßregisterinhalt in den Befehlszähler zurückgebracht wird:

$$\langle AR \rangle := \langle BZ \rangle$$
$$\langle BZ \rangle := \langle AR \rangle + 1^{\circ}.$$

Ein programmierter Stop erfolgt am Ende der Phase III. Eine eventuell geforderte Löschung des Akkumulators findet ebenfalls in dieser Phase statt.

Der Informationszustand nach Phase III ist durch

$$\langle BZ \rangle = b + 1^{\circ},\ \langle AR \rangle = a,\ \langle OR \rangle = \omega^*$$

gegeben, wobei ω^* die eventuell in Phase II bereits umgestellte Operation ist.

Bei einem Sprung mit Rückkehrabsicht findet in Phase III eine disjunktive Verknüpfung des Akkumulatorinhaltes mit dem um 1 Bit rechtsverschobenen Befehlszählerinhalt statt.

Bei Stop-Befehlen hängt die Fortsetzung der Befehlsschleife von der Bedienung ab. Eingabe-Halt- und Ausgabe-Halt-Befehle werden einer Sonderbehandlung unterworfen.

Bei bedingten Sprungbefehlen mit nicht erfüllter Bedingung folgt nach einer Wortzeit Phase I; bei unbedingten Sprungbefehlen oder bedingten Sprungbefehlen mit erfüllter Bedingung folgt die Resultatbehandlung in Phase IV.

Bei allen anderen Befehlen folgt Phase IV.

2.2.4. Phase IV

Phase IV wird eventuell erst nach einer Wartezeit W_2 zum Erreichen der Koinzidenz zwischen der im Adreßregister aufgestellten Sektornummer von a und der Trommelstellung eingeschaltet.

In dieser Phase werden der Akkumulatorinhalt und der Inhalt der Speicherzelle a entsprechend der Operationsanweisung miteinander verknüpft, oder es wird der entsprechende Transport ausgeführt. Anschließend folgt die Resultatbehandlung, die sich stets auf den Akkumulatorinhalt bezieht.

Am Ende der Phase IV liegt folgender Informationszustand vor:

$$\langle BZ \rangle = b + 1^{\circ},\ \langle AR \rangle = a,\ \langle OR \rangle = \omega^*,\ \langle Ac \rangle = R.$$

Je nach Befehlstyp kann die Operation 1 mal bis 32 mal ausgeführt werden. Für jede Ausführung wird eine Wortzeit benötigt.

Bei einem bestimmten Befehlstyp (markierte Einzelbefehle) wird zusätzlich einer von drei möglichen Vergleichszuständen erzeugt.

Ist im vorhergehenden Befehl ein Überlauf (Überschreitung des Zahlenbereiches) erfolgt, so wird der Ablauf der Befehlsschleife vor der Phase IV unterbrochen, wenn der abzuarbeitende Befehl kein Überlaufsprung ist. (Der Überlaufsprung wird dann bereits in Phase II in einen unbedingten Sprung umgestellt.) Überlauf wird nur bei Addition, Subtraktion, bedingter Addition bzw. Subtraktion als Einzelbefehl oder als markierter Einzelbefehl ohne Resultatsbehandlung sowie bei Befehlen, die in solche umgestellt werden, eingeschaltet.

Überlauf bedeutet, daß der Übertrag von z_1 nach z_0 vom Übertrag von z_0 zur nicht mehr vorhandenen Maschinenstelle verschieden ist.

2.3. Speicherorganisation

Die $4096 = 2^{12}$ Speicherzellen sind auf der Magnettrommel in 128 Spuren mit je 32 Zellen angeordnet. Aus der Trommelfrequenz von 18 000 Umdrehungen pro Minute ergeben sich damit Zugriffszeiten zwischen 0 und 3,33 ms; die mittlere Zugriffszeit beträgt also ca. 1,67 ms, die Wortzeit, d. h. die Zeit, in der sich die Trommel um eine Wortlänge weiterdreht, ca. 0,1 ms.

Jede Speicherzelle kann durch die Angabe ihrer Spurnummer und ihrer Sektornummer identifiziert werden. Ihre Adresse kann demnach (dezimal) als

$$a = 32 \cdot \alpha + \beta$$
$$(\alpha = 0, 1, \ldots, 127),$$
$$(\beta = 0, 1, \ldots, 31)$$

dargestellt werden. Damit entfallen von den 12 Dualstellen der Adresse 7 Bits auf die Spurnummer, 5 Bits auf die Sektornummer (Abb. 4).

Aus Zweckmäßigkeitsgründen empfiehlt sich für die Programmierung die Zusammenfassung von Spur- und Sektornummer zu einer einheitlichen vierstelligen Oktalzahl $a_1\ a_2\ a_3\ a_4$. Dabei gehören die beiden niedrigsten Bits der Oktalziffer a_3 zur Sektorangabe, ihr höchstes Bit gehört zur Spurangabe.

Die oktale Numerierung läuft von 0000 bis 7777. Die Adresse der ersten Zelle jeder Spur hat eine der beiden Formen XX00 oder XX40.

Um eine günstige Befehlsfrequenz zu erreichen, wurde eine von der geometrischen Anordnung auf der Trommel abweichende Numerierung der Speicherzellen als deren natürliche Reihenfolge festgelegt. Sie berücksichtigt den zeitlichen Ablauf der Befehlsschleife. Im allgemeinen werden für die Abarbeitung eines Befehles vier Wortzeiten benötigt. Deshalb könnte bei der Numerierung im Viererschnitt vorgegangen werden, ohne daß Wartezeiten auftreten. Dabei wäre jedoch nur ein Viertel aller Speicherzellen ohne zusätzliche Sprungbefehle erreichbar. Aus diesem Grund wurde tatsächlich ein Fünferschnitt gewählt, so daß zwar eine Wortzeit mehr benötigt wird, aber der genannte Nachteil nicht auftritt; gleichzeitig stehen zwei Speicherzellen für Operanden ohne zusätzliche Wartezeit zur Verfügung. Die Verteilung der Sektornummern innerhalb einer Spur ist in Abb. 5 angegeben.

Diese Speicherorganisation gibt die Möglichkeit zeitoptimaler Programmierung. Zum Beispiel möge der erste Befehl eines Programmes aus einer Speicherzelle mit der (oktalen) Sektornummer 0 entnommen werden (Phase II). Steht im Adreßteil dieses Befehls eine der Sektornummern 32 oder 7, so kann der nächste Befehl, der die Sektornummer 1 hat, ohne zusätzliche Wartezeit entnommen werden. Steht jedoch im Adreßteil des ersten Befehls z. B. die Sektornummer 16, so kann der nächste Befehl erst eine ganze Trommelumdrehung später entnommen werden.

Allgemein sind bei oktaler Rechnung von einem Befehl aus der Zelle a die Operandenzellen $a+k+32$ und $a+k+7$ ohne zusätzliche Wartezeit für die Entnahme des nächsten Befehls greifbar, wobei k ein ganzzahliges Vielfaches von 40 ist. So sind z. B. für einen Befehl aus 5613 die Operandenzellen 0005, 0022, 0045, 0062, ..., 7745, 7762 zeitoptimal greifbar.

Gruppenbefehle (s. 4. Befehlssystem) enden vor der Sektornummer, die im Adreßteil des Befehls steht, so daß für einen Gruppenbefehl aus der Zelle a die Operandenzellen $a+k+32$, $a+k+7$, $a+k+24$ zeitoptimal sind.

3. Maschinenwort

3.1. Interne Darstellung

Das interne Maschinenwort besteht aus 33 Bits und kann gleichermaßen zur Darstellung von Zahlen, Befehlen oder Alpha-Zeichen dienen. Die Art der Interpretation wird durch das Programm bestimmt. Dabei sind auch verschiedenartige Interpretationen innerhalb eines Wortes möglich.

Die 33 Bits des Maschinenwortes werden von links nach rechts als $z_0, z_1, \dots, z_{32}$ bezeichnet (Abb. 6). Jedes Bit kann mit 0 oder 1 besetzt sein, so daß es unabhängig von der Interpretation 2^{33} verschiedene Wörter gibt. In jeder Speicherzelle kann genau ein Wort untergebracht werden.

3.1.1. Dualzahlen

Bei der Ausführung der verdrahteten Rechenbefehle (Addition, Subtraktion) wird das interne Wort vom Rechner als Dualzahl in Festkommadarstellung interpretiert. Die Stellung des Kommas wird dabei nicht festgelegt; beispielsweise denke man es sich zwischen z_0 und z_1. Das Bit z_0 enthält das Vorzeichen.

Positive Zahlen z haben dann die Darstellung

$$z = \sum_{i=1}^{32} z_i \cdot 2^{-i}.$$

Negative Zahlen $\tilde{z}$ werden als Zweierkomplement dargestellt:

$$\tilde{z} = 2 - z$$

Dabei werden alle Bits vor dem niedrigsten mit 1 besetzten negiert (Abb. 7).

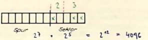


Abb. 4

dezimal

00	13	26	07	20	01	14	27	08	21	02	15	28	09	22	03	16	29	10	23	04	17	30	11	24	05	18	31	12	25	06	19
00	15	32	07	24	01	16	33	10	25	02	17	34	11	26	03	20	35	12	27	04	21	36	13	30	05	22	37	14	31	06	23

oktal

mod. 40

Abb. 5

□ = Zeitoptimale Adressen, wenn mit 00 begonnen wird.

z_0	z_1	z_2	z_3	z_4	z_5	z_6	z_7	z_8	z_9	z_{10}	z_{11}	z_{12}	z_{13}	z_{14}	z_{15}	z_{16}	z_{17}	z_{18}	z_{19}	z_{20}	z_{21}	z_{22}	z_{23}	z_{24}	z_{25}	z_{26}	z_{27}	z_{28}	z_{29}	z_{30}	z_{31}	z_{32}

Abb. 6

2^{30} verschiedene Worte

Bit	z_0	z_1	z_2	z_3	z_4	z_5	z_6	z_7	z_8	z_9	z_{10}	z_{11}	z_{12}	z_{13}	z_{14}	z_{15}	z_{16}	z_{17}	z_{18}	z_{19}	z_{20}	z_{21}	z_{22}	z_{23}	z_{24}	z_{25}	z_{26}	z_{27}	z_{28}	z_{29}	z_{30}	z_{31}	z_{32}
z	0	1	0	1	1	1	0	0	0	0	1	0	1	1	0	1	1	0	0	0	1	1	0	1	1	1	1	0	1	0	0	0	0
-z	1	0	1	0	0	0	1	1	1	1	0	1	0	0	1	0	0	1	1	1	0	0	1	0	0	0	0	1	1	0	0	0	0

wird vertauscht | bleibt erhalten } Negative Zahlen

Abb. 7

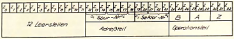


Abb. 8

Triade	t_{10}			t_9			t_8			t_7			t_6			t_5			t_4			t_3			t_2			t_1			t_0		
Bit	z_0	z_1	z_2	z_3	z_4	z_5	z_6	z_7	z_8	z_9	z_{10}	z_{11}	z_{12}	z_{13}	z_{14}	z_{15}	z_{16}	z_{17}	z_{18}	z_{19}	z_{20}	z_{21}	z_{22}	z_{23}	z_{24}	z_{25}	z_{26}	z_{27}	z_{28}	z_{29}	z_{30}	z_{31}	z_{32}
Dualz.	0	1	1	1	0	1	1	0	0	0	0	0	1	1	1	0	0	1	1	1	0	1	0	1	0	1	0	1	1	0	0	1	1
Oktalz.	3			5			4			0			7			1			6			5			2			6			3		

Abb. 9

Damit sind positive Zahlen durch $z_0 = 0$, negative durch $z_0 = 1$ charakterisiert. Der Zahlenbereich ist durch

$$-1 \leq z \leq 1 - 2^{-32}$$

gegeben. Die kleinste von Null verschiedene positive Zahl ist 2^{-32}; die Null selbst gilt als positiv. Die angegebene Darstellung ist umkehrbar eindeutig.

3.1.2. Befehle

Zur Darstellung eines Befehls werden nur 21 Bits eines Wortes benötigt; die ungenutzten können bei der Programmierung anderweitig verwendet werden. Das Befehlswort besteht aus Adreßteil und Operationsteil.
Der Adreßteil enthält 12 Bits zum Aufruf einer der 4096 Speicherzellen. Die einzelnen Bits werden tripelweise zu vier Oktalziffern $a_1a_2a_3a_4$ zusammengefaßt.
Der Operationsteil zerfällt im Sinne eines analytischen Befehlssystems in drei Gruppen von je drei Bits (Abb. 8):

3 Bits zur Bezeichnung des Befehlstyps B,
3 Bits zur Bezeichnung der Operationsart A,
3 Bits zur Bezeichnung des Operationszusatzes Z.

3.2. Triadendarstellung

Eine bitweise Notierung der Information außerhalb des Rechners etwa zur Eingabe wäre recht mühsam und unbequem. Deshalb wird je nach Art der Information eine besondere externe Darstellung benutzt. Auch die vom Rechner ausgegebenen Informationen erscheinen je nach ihrer Art in einer von der internen abweichenden Darstellung. Alle externen Darstellungen sind von der Interpretation durch das Programm abhängig.
Unabhängig von der inhaltlichen Bedeutung der Information können im Wort für die Eingabe jeweils 3 Bits zu einer Oktalziffer oder Triade t_i zusammengefaßt werden, so daß das Wort durch 11 Triaden dargestellt wird (Abb. 9).

4. Befehlssystem

Nach 3.1.2. besteht der Operationsteil eines Befehls aus den drei Triaden B, A, Z, die im Sinne eines analytischen Befehlscodes weitgehend selbständige Bedeutung haben. Der Operationszusatz Z setzt sich aus dem Paar R für die Resultatbehandlung und dem Löschbit L zusammen. Im folgenden wird die Bedeutung im einzelnen dargestellt.

4.1. Befehlstyp

Der Befehlstyp wird durch die Bits z_{24}, z_{25}, z_{26} angegeben; er kann einen der acht Zahlenwerte 0 bis 7 besitzen. Durch die verschiedenen Werte des Befehlstyps werden verschiedene Bedeutungen der Operationsart festgelegt.

4.1.1. Organisationsbefehl 1. Art

Organisationsbefehle 1. Art sind durch **B = 0** charakterisiert. In Abhängigkeit von der Operationsart brauchen nicht alle Phasen der Befehlsschleife durchlaufen zu werden; die Phasen I oder IV können entfallen.

4.1.2. Organisationsbefehl 2. Art

Organisationsbefehle 2. Art sind durch **B = 1** charakterisiert. Der Phasenablauf kann wie bei Organisationsbefehlen 1. Art modifiziert werden.

4.1.3. Einzelbefehl

Einzelbefehle sind durch **B = 2** charakterisiert. Die Operation bezieht sich auf eine einzige Speicherzelle.

4.1.4. Markierter Einzelbefehl

Markierte Einzelbefehle sind durch **B = 3** charakterisiert. Zusätzlich zur Wirkung des Einzelbefehls wird ein Vergleichszustand erzeugt. Welcher Vergleichszustand erzeugt wird, ist vom Resultat eines Vergleichs abhängig.
Für die Darstellung wird der Inhalt des Vergleicher-Flip-Flop-Paares als zweistellige Dualzahl aufgefaßt.

Verglichen wird der Akkumulatorinhalt nach Phase III mit einer Größe v. Dabei ist v = ⟨a⟩ für Operationsarten $A \leq 3$ mit dem Adreßteil a und es ist v = 0 für Operationsarten $A \geq 4$. Der Akkumulatorinhalt selbst ist zum Zeitpunkt des Vergleichs für alle markierten Einzelbefehle mit Löschanweisung gleich Null.

Das Flip-Flop 1 bleibt eingeschaltet, wenn $\langle Ac\rangle \leq v$ gilt; für $\langle Ac\rangle \geq v$ bleibt Flip-Flop 2 eingeschaltet. Gilt $\langle Ac\rangle = v$, so bleiben beide Flip-Flop eingeschaltet; der Wert des Vergleichszustandes ist gleich 3.

Im Mechanismus des Vergleichersetzens werden zunächst am Ende von Phase III beide Flip-Flop eingeschaltet; der Vergleicher 3 wird gesetzt. In Phase IV findet nur bei Ungleichheit von ⟨Ac⟩ und v eine Korrektur des Vergleichers statt, wobei beide Flip-Flop stets entgegengesetzt geschaltet werden.

Das Symbol V (a, b) bedeutet im weiteren, daß a als ⟨Ac⟩ mit b verglichen wird.

4.1.5. Gruppenbefehl

Gruppenbefehle (G-Befehle) sind durch B = 4 charakterisiert. Der Gruppenbefehl beginnt wie ein Einzelbefehl, d. h. die Phasen I bis III werden wie bei diesem ausgeführt. Die Phase IV bezieht sich dagegen auf alle Zellen der Spur, zu der die im Adreßteil genannte Zelle gehört; sie wird also 32mal durchlaufen. Die Zellen dieser Spur werden dabei in der Reihenfolge ihrer geometrischen Anordnung auf der Trommel bearbeitet.

4.1.6. Verkürzter Gruppenbefehl

Verkürzte Gruppenbefehle (GX-Befehle) sind durch B = 5 charakterisiert. Der verkürzte Gruppenbefehl beginnt wie ein Gruppenbefehl, hat aber eine zusätzliche Abbruchbedingung (X-Bedingung).

Die zusätzliche Abbruchbedingung besteht in der Negativität des Akkumulatorinhaltes; bei $R \neq 1$ wird z_0, bei R = 1 wird z_{32} als Vorzeichen interpretiert. Nachdem diese Abbruchbedingung erfüllt ist, werden Operation und Resultatbehandlung noch mit einer Sektornummer ausgeführt. Der Abbruch erfolgt spätestens nach voller Ausführung des Gruppenbefehls.

Der nächste Befehl wird nach erfolgtem X-Abbruch ohne Wartezeit W_1 sofort entnommen. Demnach wird die Spur für diese Befehlsentnahme durch den Inhalt des Befehlszählers bestimmt, während die Sektornummer in der zweiten Wortzeit nach Abbruch des GX-Befehles liegt; in der ersten Wortzeit nach dem Abbruch erfolgt Phase I. (Abb. 10) Wird die X-Bedingung nicht realisiert, so erfolgen Abbruch und Fortsetzung wie beim G-Befehl.

Werden normale und zusätzliche Abbruchbedingungen gleichzeitig erfüllt, geschieht die Entnahme des nächsten Befehls so, als ob die normale Abbruchbedingung den Vorrang hätte. Genauer ist der Wirkungsmechanismus aus Abb. 10 zu entnehmen. Aus dem Entnahmemechanismus, für den auf den GX-Befehl folgenden Befehl, ergeben sich einige weitere Konsequenzen für die Programmfortsetzung, wenn die X-Bedingung zum Abbruch führt. Wurde der X-Befehl aus der Zelle b entnommen, so hat der Befehlszähler nach Phase III des nächsten Befehls den Inhalt b + 2. Ist der nach dem GX-Befehl entnommene Befehl ein tatsächlich auszuführender Sprungbefehl (Sprung ohne Bedingung oder bedingter Sprung mit erfüllter Bedingung), so wird dieser Befehlszählerinhalt nicht ausgewertet. Wird kein Sprung ausgeführt, so wird der nunmehr folgende Befehl aus Zelle b + 2 entnommen.

4.1.7. Wiederholungsbefehl

Wiederholungsbefehle (W-Befehle) sind durch B = 6 charakterisiert. Der Wiederholungsbefehl wirkt wie ein Gruppenbefehl mit anderer Abbruchbedingung für die Wiederholung. Er beginnt bei der im Adreßteil angegebenen Speicherzelle und endet bei derjenigen mit der (oktalen) Sektornummer 23; ein Wiederholungsbefehl mit einer Adresse mit der Sektornummer 0 wirkt wie ein Gruppenbefehl mit derselben Adresse. Durch einen Wiederholungsbefehl kann also eine n-malige Ausführung von Operationen mit $1 \leq n \leq 32$ realisiert werden.

4.1.8. Verkürzter Wiederholungsbefehl

Verkürzte Wiederholungsbefehle (WX-Befehle) sind durch B = 7 charakterisiert. Der verkürzte Wiederholungsbefehl beginnt wie ein Wiederholungsbefehl, hat aber dieselbe zusätzliche Abbruchbedingung wie der verkürzte Gruppenbefehl. Der Abbruch erfolgt spätestens nach voller Ausführung des Wiederholungsbefehls.

Die Programmabarbeitung wird wie beim verkürzten Gruppenbefehl fortgesetzt. (Abb. 10)

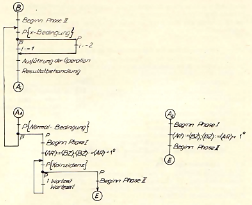


Abb. 10

4.2. Operationszusatz

Der Operationszusatz Z steht in den Bits z_{30}, z_{31}, z_{32}; er hat also einen der acht Zahlenwerte 0 bis 7. Das Bitpaar z_{30}, z_{31} besitzt die Bedeutung der Resultatbehandlung R, das Bit z_{32} die Bedeutung der Löschanweisung L.

4.2.1. Resultatbehandlung

Die Resultatbehandlung geschieht stets nach Ausführung der Operation in Phase IV und bezieht sich auf den Inhalt des Akkumulators. Bei einigen Organisationsbefehlen wird dieser Teil der Befehlsschleife nicht durchlaufen; die Resultatbehandlung ist dann wirkungslos (Abb. 3). Es gibt vier Möglichkeiten der Resultatbehandlung, die den Zahlenwerten 0 bis 3 entsprechen.

Für R = 0 wird **keine Resultatbehandlung** durchgeführt; das Bitpaar z_{30}, z_{31} ist wirkungslos;

$$\langle Ac \rangle := \langle Ac \rangle$$

(In diesem Abschnitt wird nur die Wirkung der Resultatbehandlung beschrieben und für diese ein Symbol definiert.)

Die Resultatbehandlung R = 1 bedeutet **Rechtsverschiebung.** Das Resultat wird um ein Bit nach rechts verschoben; das rechts überlaufende Bit geht verloren, z_0 wird mit 0 besetzt:

$$\langle Ac \rangle := \tfrac{1}{2} \cdot \langle Ac \rangle \overset{\text{def}}{=} V_r^1 \left(\langle Ac \rangle\right).$$

Linksverschiebung ist durch R = 2 charakterisiert. Das Resultat wird um ein Bit nach links verschoben; das links überlaufende Bit geht verloren, z_{32} wird mit 0 besetzt:

$$\langle Ac \rangle := 2 \cdot \langle Ac \rangle \overset{\text{def}}{=} V_l^1 \left(\langle Ac \rangle\right).$$

Bei R = 3 wird eine **zyklische Linksverschiebung** durchgeführt. Das Resultat wird um ein Bit nach links verschoben; das links überlaufende Bit wird in die Stelle z_{32} übertragen:

$$\langle Ac \rangle := 2 \cdot \langle Ac \rangle + z_0 \cdot 2^{-32} \overset{\text{def}}{=} V_{zl}^1 \left(\langle Ac \rangle\right).$$

4.2.2. Löschbit

Das Löschbit wird in Phase III wirksam und bezieht sich auf den Inhalt des Akkumulators. Es gibt zwei Möglichkeiten. (Abb. 3)

Hat das Löschbit den Wert L = 0, so ist es wirkungslos. Für L = 1 wird der Akkumulatorinhalt in Phase III des Befehls, der die Löschanweisung enthält, gelöscht, alle Bits werden zu Null.

4.2.3. Zusammenfassung von Resultatbehandlung und Löschanweisung

Aus den vier Möglichkeiten für die Resultatbehandlung und den zwei Möglichkeiten für die Löschanweisung ergeben sich acht mögliche Werte für den Operationszusatz mit ihren verschiedenen Bedeutungen:

Z	Bedeutung
0	keine Löschung, keine Resultatbehandlung
1	Löschen, keine Resultatbehandlung
2	keine Löschung, Rechtsverschiebung
3	Löschen, Rechtsverschiebung
4	keine Löschung, Linksverschiebung
5	Löschen, Linksverschiebung
6	keine Löschung, zyklische Linksverschiebung
7	Löschen, zyklische Linksverschiebung

4.3. Operationsart

Die Bedeutung der Triade für die Operationsart ist vom Befehlstyp abhängig; sie ist jeweils verschieden für Organisationsbefehle 1. Art (B = 0), Organisationsbefehle 2. Art (B = 1) und nichtorganisatorische Befehle (B ≧ 2).
Es gibt jeweils acht Möglichkeiten.

4.3.1. Nichtorganisatorische Befehle

Nichtorganisatorische Befehle sind durch B ≧ 2 charakterisiert. Bei G-, GX-, W- und WX-Befehlen wird Phase IV entsprechend oft wiederholt. Zusätzlich zur eigentlichen Operation werden die übrigen Phasen und die Resultatbehandlung ausgeführt; bei markierten Einzelbefehlen kommt dazu noch das Vergleichersetzen.

Da der Operationsart jeweils sechs Befehlstypen angehören, die mit acht verschiedenen Operationszusätzen auftreten können, ergeben sich für jede Operationsart 48 formal verschiedene Operationsteile. Im allgemeinen liefern einige davon identische Resultate, die dabei jedoch auf unterschiedliche Weise und mit unterschiedlichem Zeitbedarf entstehen können.

4.3.1.1. Konjuktion im Akkumulator

Die Operation ist charakterisiert durch A = 0. Sie besteht in der bitweisen Konjunktion des Akkumulatorinhaltes mit dem Inhalt der im Adreßteil angegebenen Speicherzelle; das Resultat steht im Akkumulator:

$$\langle Ac\rangle := \langle Ac\rangle \wedge \langle a\rangle\,.$$

Beispiel:

⟨Ac⟩	=	000	000	000	000	111	111	111	111	000	000	000
⟨a⟩	=	000	000	000	000	011	010	000	100	010	010	010
⟨Ac⟩ ∧ ⟨a⟩	=	000	000	000	000	011	010	000	100	000	000	000

Überlauf kann nicht erzeugt werden.

Alle Befehle mit Löschanweisung liefern als Ergebnis den Akkumulatorinhalt Null. Bei den Operationen 301, 303, 305, 307 wird zusätzlich gemäß V (0, ⟨a⟩) ein Vergleicher gesetzt. Weil der Akkumulatorinhalt Null bleibt, kann bei GX- und WX-Befehlen die X-Bedingung nicht wirksam werden, so daß GX- und G-Befehl sowie WX- und W-Befehl jeweils identisch ablaufen; sie haben in Abhängigkeit vom Adreßteil unterschiedlichen Zeitbedarf. (Abb. 11)

Die 20 formal verschiedenen Operationsteile 201, 203, 205, 207, 401, 403, 405, 407, 501, 503, 505, 507, 601, 603, 605, 607, 701, 703, 705, 707 sowie die vier Operationsteile 301, 303, 305, 307 liefern also jeweils das gleiche Resultat. Damit verringert sich die Anzahl der verschiedenen Resultate von 48 möglichen auf 26.

4.3.2.1. Addition

Die Operation ist charakterisiert durch A = 1. Sie addiert zum Akkumulatorinhalt den Inhalt der im Adreßteil angegebenen Speicherzelle; das Resultat steht im Akkumulator:

$$\langle Ac\rangle := \langle Ac\rangle + \langle a\rangle\,.$$

Ausgangszustand: <OR> = BOZ, <Ac> = R-1
<AR> = b
<BZ> = a

Entspr. Abb. 5:
$15_8 + 15_8 = 32_8$
$32_8 + 15_8 = 07_8$

0 + 15 mod 40
= nächste Zelle

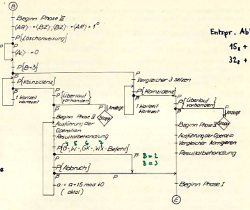


Abb. 11

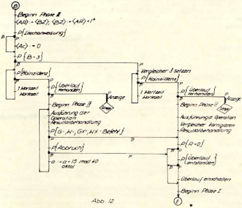


Abb. 12

Beispiel:

(Ac)	=	000	000	000	000	111	111	111	111	000	000	000
(a)	=	000	000	000	000	011	010	000	100	010	010	010
(Ac) + (a)	=	000	000	000	001	011	001	000	011	010	010	010

Überlauf kann bei Einzelbefehlen und markierten Einzelbefehlen eingeschaltet werden, wenn keine Resultatbehandlung verlangt ist; das betrifft nur die zwei Operationsteile 210 und 310. (Abb. 12) B = 2 , B = 3

Alle 48 Operationsteile haben paarweise verschiedene Wirkung. Der Operationsteil 211 hat die Wirkung eines Lesebefehls:

$$\langle Ac \rangle := \langle a \rangle ;$$

der Operationsteil 210 bewirkt eine Addition im Festkomma:

$$\langle Ac \rangle := \langle Ac \rangle + \langle a \rangle .$$

4.3.1.3. Subtraktion

Die Operation ist charakterisiert durch A = 2. Vom Inhalt des Akkumulators wird der Inhalt der im Adreßteil angegebenen Speicherzelle subtrahiert; das Resultat steht im Akkumulator:

$$\langle Ac \rangle := \langle Ac \rangle - \langle a \rangle .$$

Beispiel:

⟨Ac⟩	=	000	000	000	000	111	111	111	111	000	000	000
⟨a⟩	=	000	000	000	000	011	010	000	100	010	010	010
⟨Ac⟩ − ⟨a⟩	=	000	000	000	000	100	101	111	010	101	101	110

Überlauf kann bei Einzelbefehlen und bei markierten Einzelbefehlen eingeschaltet werden, wenn keine Resultatbehandlung verlangt ist; das sind nur die Operationsteile 220, 221, 320 und 321. (Abb. 12)

Alle 48 verschiedenen Operationsteile haben paarweise verschiedene Wirkungen. Der Operationsteil 221 hat die Wirkung eines Lesebefehls für das Komplement:

$$\langle Ac \rangle := -\langle a \rangle ;$$

der Operationsteil 220 bewirkt eine Subtraktion im Festkomma:

$$\langle Ac \rangle := \langle Ac \rangle - \langle a \rangle .$$

4.3.1.4. Bedingte Addition (Subtraktion)

Die Operation ist durch A = 3 charakterisiert; die Operationsteile 330, 331 bewirken jedoch einen anderen Ablauf.

Der Ablauf hängt von der Resultatbehandlung ab; er ist bei Rechtsverschiebung gegenüber anderer Resultatbehandlung verschieden.

Bei Rechtsverschiebung wird z_{32} als Vorzeichenbit interpretiert. Ist der Akkumulatorinhalt negativ, wird dazu der Inhalt der im Adreßteil angegebenen Speicherzelle additiert; ist der Akkumulatorinhalt positiv, bleibt er ungeändert:

$$\text{sgn}\,\langle Ac \rangle := z_{32}$$

$$\langle Ac \rangle := \begin{cases} \langle Ac \rangle + \langle a \rangle & \text{für sgn}\,\langle Ac \rangle = 1 \\ \langle Ac \rangle & \text{für sgn}\,\langle Ac \rangle = 0 . \end{cases}$$

Beispiel:

⟨Ac⟩	=	000	000	000	000	111	111	111	111	000	000	00^1_0
⟨a⟩	=	000	000	000	000	011	010	000	100	010	010	010
⟨Ac⟩(±)⟨a⟩	=	000	000	000	00^1_0	011 111	010 111	000 111	011 111	010 000	010 000	011 000

Bei anderer Resultatbehandlung wird z_0 als Vorzeichenbit interpretiert. Ist der Akkumulatorinhalt negativ, wird dazu der Inhalt der im Adreßteil angegebenen Speicherzelle addiert; ist der Akkumulatorinhalt positiv, wird der Inhalt dieser Speicherzelle subtrahiert:

$$\text{sgn}\,\langle Ac \rangle := z_0$$

$$\langle Ac \rangle := \begin{cases} \langle Ac \rangle + \langle a \rangle & \text{für sgn}\,\langle Ac \rangle = 1 \\ \langle Ac \rangle - \langle a \rangle & \text{für sgn}\,\langle Ac \rangle = 0 . \end{cases}$$

Beispiel:

⟨Ac⟩	=	${}^1_000$	000	000	000	111	111	111	111	000	000	000
⟨a⟩	=	000	000	000	000	011	010	000	100	010	010	010
⟨Ac⟩ ± ⟨a⟩	=	${}^1_000$	000	000	00^1_0	011 100	010 101	000 111	011 010	010 101	010 101	010 110

Überlauf kann bei Einzelbefehlen und markierten Einzelbefehlen eingeschaltet werden, wenn keine Resultatbehandlung verlangt ist; das sind nur die Operationsteile 230 und 231. (Abb. 12)

Die 46 verschiedenen Operationsteile liefern nur 42 nach außen verschiedene Wirkungen.

Die Operationen 233, 433, 533, 633, 733 löschen nur den Akkumulator. Sie enthalten alle die Löschanweisung und verlangen Rechtsverschiebung, so daß der Akkumulatorinhalt vor Phase IV Null ist und z_{32} als Vorzeichenbit interpretiert wird.

Da $z_{32} = 0$ ist, wird in Phase IV der Akkumulatorinhalt nicht mehr verändert; auch die Rechtsverschiebung ist wegen ⟨Ac⟩ = 0 wirkungslos. Bei den Operationen

533, 733 tritt deshalb nie ein X-Abbruch ein. Die Operation 333 besitzt die oben beschriebene Wirkung und setzt zusätzlich einen Vergleicher; sie ist mit 301 wirkungsgleich.

Die Operationen 231, 235, 237, 335, 337 interpretieren bei gelöschtem Akkumulatorinhalt z_0 als Vorzeichenbit, bewirken also eine Subtraktion und sind demzufolge in dieser Reihenfolge mit den Operationen 221, 225, 227, 325, 327 identisch.

4.3.1.5. Reines Vergleichersetzen

Bei den Operationsteilen 330, 331 wird die Ausführung der Operation in Phase IV unterdrückt; es wird nur die Löschanweisung ausgeführt und ein Vergleicher gesetzt. Überlauf kann nicht erzeugt werden. (Abb. 13)

Die Operation 331 gleicht in ihrer Wirkung der Operation 301.

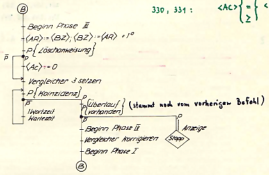


Abb 13

4.3.1.6. Transport

Die Operation ist durch A = 4 charakterisiert. Der Akkumulatorinhalt wird zu der im Adreßteil angegebenen Speicherzelle transportiert; der Akkumulatorinhalt bleibt (bis auf eventuelle Löschung und Resultatbehandlung) erhalten:

$$\langle a \rangle := \langle Ac \rangle .$$

Überlauf kann nicht erzeugt werden. (Abb. 11)

Die 48 verschiedenen Operationsteile ergeben nur 28 nach außen verschiedene Wirkungen.

Die Operationen 241, 243, 245, 247 löschen den Akkumulatorinhalt und den Inhalt der im Adreßteil angegebenen Speicherzelle. Die Operationen 341, 343, 345, 347 setzen zusätzlich zur oben beschriebenen Wirkung einen Vergleicher gemäß V (0, 0), also den Vergleicher 3.

Die Operationen 441, 443, 445, 447 löschen den Akkumulatorinhalt und die Inhalte aller Zellen der durch den Adreßteil bestimmten Spur. Bei den Operationen 541, 543, 545, 547 kann wegen der Löschanweisung kein X-Abbruch eintreten, so daß ihre Wirkung mit der von 441 identisch ist.

Die Operationen 641, 643, 645, 647 löschen analog zur Operation 441 den Akkumulatorinhalt und den Inhalt aller Zellen, die in der betreffenden Spur auf die durch den Adreßteil angegebene Speicherzelle bis zur Nullstellung der Trommel folgen.

Bei den Operationen 741, 743, 745, 747 kann wegen der Löschanweisung analog zur Operation 541 kein X-Abbruch eintreten, so daß ihre Wirkung mit der von 641 identisch ist.

Der Operationsteil 240 speichert den Akkumulatorinhalt nach der im Adreßteil angegebenen Speicherzelle ab, ohne den Akkumulatorinhalt zu verändern:

$$\langle a \rangle := \langle Ac \rangle .$$

4.3.1.7. Konjunktion im Speicher

Die Operation ist durch $A = 5$ und $L = 0$ charakterisiert. (Für $L = 1$ wäre das Ergebnis der Konjunktion stets Null.)

Sie besteht in der bitweisen Konjunktion des Akkumulatorinhaltes mit dem Inhalt der im Adreßteil angegebenen Speicherzelle. Das Resultat steht in dieser Speicherzelle; der Akkumulatorinhalt bleibt (bis auf eine eventuelle Resultatbehandlung) erhalten:

$$\langle a \rangle := \langle Ac \rangle \wedge \langle a \rangle .$$

Überlauf kann nicht erzeugt werden. (Abb. 11)

Alle 24 verschiedenen Operationsteile haben paarweise verschiedene Wirkung.

Die Wirkung des Operationsteils 250 beschränkt sich auf

$$\langle a \rangle := \langle Ac \rangle \wedge \langle a \rangle ;$$

der Akkumulatorinhalt wird nicht verändert.

4.3.1.8. Konjunktion mit negiertem Akkumulatorinhalt

Die Operation ist durch $A = 6$ charakterisiert. Sie besteht in der bitweisen Konjunktion des negierten Akkumulatorinhaltes mit dem Inhalt der im Adreßteil angegebenen Speicherzelle. Das Resultat steht in dieser Speicherzelle; der Akkumulator bleibt (bis auf eventuelle Löschung und Resultatbehandlung) erhalten:

$$\langle a \rangle := \overline{\langle Ac \rangle} \wedge \langle a \rangle .$$

Beispiel:

$\langle Ac \rangle$	= 000	000	000	000	111	111	111	111	000	000	000
$\overline{\langle Ac \rangle}$	= 111	111	111	111	000	000	000	000	111	111	111
$\langle a \rangle$	= 000	000	000	000	011	010	000	100	010	010	010
$\overline{\langle Ac \rangle} \wedge \langle a \rangle$	= 000	000	000	000	000	000	000	000	010	010	010

Überlauf kann nicht erzeugt werden. (Abb. 11)

Alle Operationsteile mit Löschanweisung liefern einen Operanden $\langle Ac \rangle$, in dem alle Bits mit 1 besetzt sind, womit die Konjunktion als Operation wirkungslos bleibt. Damit bewirkt der Befehl lediglich das Löschen des Akkumulatorinhaltes.

Bei den Operationen 361, 363, 365, 367 wird wegen $\langle Ac \rangle = 0$ und $A \geq 4$ zusätzlich gemäß V (0, 0) der Vergleicher 3 gesetzt. Weil der Akkumulatorinhalt Null, kann bei GX- und WX-Befehlen die X-Bedingung nicht wirksam werden, so daß auch die Entnahme des nächsten Befehls in natürlicher Reihenfolge erfolgt.

Damit ergeben die 48 formal verschiedenen Operationsteile nur 26 verschiedene Wirkungen.

4.3.1.9. Bedingte Speicherlöschung

Die Operation ist durch $A = 7$ und $L = 0$ charakterisiert. (Für $L = 1$ wäre das Resultat gleich wie bei $A = 4$.)

Der Ablauf des Befehls hängt von der Resultatbehandlung ab; er ist bei Rechtsverschiebung unterschiedlich gegenüber anderer Resultatbehandlung. Bei Rechtsverschiebung wird z_{32} als Vorzeichenbit interpretiert, bei anderer Resultatbehandlung das Bit z_0.

Bei positivem Akkumulatorinhalt wird die im Adreßteil angegebene Speicherzelle gelöscht, bei negativem Akkumulatorinhalt wird der Inhalt dieser Speicherzelle nicht verändert; der Akkumulatorinhalt bleibt (bis auf eine eventuelle Resultatbehandlung) erhalten:

$$\mathrm{sgn}\,\langle Ac \rangle := \begin{cases} z_{32} & \text{für } Z = 2 \\ z_0 & \text{für } Z \neq 2 \end{cases}$$

$$\langle a \rangle := \begin{cases} 0 & \text{für sgn } \langle Ac \rangle = 0 \\ \langle a \rangle & \text{für sgn } \langle Ac \rangle = 1 . \end{cases}$$

Überlauf kann nicht erzeugt werden. (Abb. 10)

Bei bedingter Speicherlöschung als G-, GX-, W- oder WX-Befehl mit Linksverschiebung oder zyklischer Linksverschiebung wird das Bit z_0 zweimal und erst danach $z_1, z_2, \ldots$ als Vorzeichenbit interpretiert.

Alle 24 unterschiedlichen Operationsteile haben verschiedene Wirkungen. Bei den Operationen 370, 372, 374, 376 wird wegen $A \geq 4$ der Vergleicher gemäß V ($\langle Ac \rangle$, 0) gesetzt.

4.3.1.10. R-Operation

Die Operation ist durch die Bedingungen A = 5, L = 1 oder A = 7, L = 1 charakterisiert. Der Ablauf der Befehlsschleife ist modifiziert. In Phase II entsteht der Inhalt des Befehlszählers als Resultat der disjunktiven Verknüpfung des Akkumulatorinhaltes mit dem Adreßteil a des aufgerufenen Befehls; das Bit z_{27} wird im Operationsregister gelöscht, so daß die Operation bei A = 5 in eine Addition, bei A = 7 in eine bedingte Addition umgestellt wird. Der weitere Ablauf erfolgt entsprechend der in Phase II entstandenen Situation ohne zusätzliche Besonderheiten. (Abb. 14)

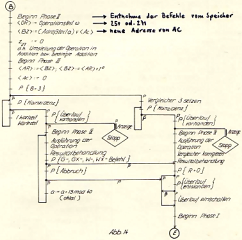


Abb. 14

Überlauf kann demnach nur bei der Operation 271 eingeschaltet werden. Hat der vorhergehende Befehl den Akkumulatorinhalt Null hinterlassen, so ist für A = 5 die Wirkung mit der von A = 1, für A = 7 die Wirkung mit der von A = 3 identisch. Bei den Operationen 351, 353, 355, 357, 371, 373, 375, 377 wird der Vergleicher gemäß V (0, (a*)) gesetzt, wobei a* die durch die disjunktive Verknüpfung in Phase II entstandene Adresse ist. Neue Adresse a* = a V Adreßteil von <AC>

Zweck: Abruf d. Inhalts einer Zelle, deren Nr. im AC steht.

Die Operationsteile 273, 473, 573, 673, 773 werden in bedingte Addition mit Rechtsverschiebung umgestellt. Da wegen der Löschung des Akkumulatorinhalts sgn (Ac) = 0 ist, wird in Phase IV keine Operation ausgeführt. Weil der Akkumulatorinhalt Null bleibt, kann bei den Operationen 573, 773 die X-Bedingung nicht wirksam werden. Deshalb bewirken die genannten Operationsteile lediglich eine Löschung des Akkumulatorinhaltes. Der Operationsteil 373 wirkt wie 301.

Die 48 formal verschiedenen Operationen ergeben also nur 44 verschiedene Wirkungen.

4.3.2. Organisationsbefehle 1. Art (Änderungen im Programmablauf)

Organisationsbefehle 1. Art sind durch B = 0 charakterisiert. α 0AZ

Sowohl bei Organisationsbefehlen 1. Art als auch bei Organisationsbefehlen 2. Art ist der Phasenablauf innerhalb der Befehlsschleife modifiziert. Für jede Operationsart ergeben sich 8 formal verschiedene Operationsteile, von denen jedoch einige nach außen dasselbe Resultat liefern können.

4.3.2.1. Unbedingter Sprung

α 0DZ

Der Befehl ist durch A = 0 charakterisiert. Die Programmabarbeitung wird mit dem Befehl, dessen Adresse im Adreßteil des Sprungbefehls angegeben ist, fortgesetzt; die

natürliche Reihenfolge der Befehlsabarbeitung wird also durchbrochen. Da die Adresse des nächsten Befehls bereits nach Phase III im Adreßregister steht, braucht die folgende Phase I nicht durchlaufen zu werden. (Abb. 15)

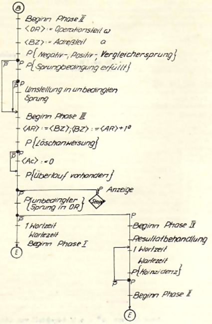


Abb. 15

Der Akkumulatorinhalt wird nur der Löschanweisung und der Resultatbehandlung unterworfen.

Die Operationsteile 001, 003, 005, 007 haben dieselbe Wirkung.

4.3.2.2. **Eingabe-Halt 1. Art**

Der Befehl ist durch A = 1 charakterisiert. Die Zentraleinheit wartet auf die Eingabe eines Zeichens vom 1-Zeichen-Puffer eines Lochbandlesers, über die Tastatur oder über das Schreibwerk. Das betreffende Eingabegerät muß vorher durch einen Anwahlbefehl bestimmt werden. Die Bits des einzugebenden Zeichens werden disjunktiv mit dem Inhalt des Adreßregisters verknüpft; bei Lochbandeingabe beginnt außerdem das Lesen des nächsten Zeichens vom Lochband und seine Speicherung im Puffer.

Beim weiteren Ablauf sind für jeden Code zwei disjunktive Klassen von Zeichen zu unterscheiden. Bei Eingabe eines Zeichens der einen Klasse wird durch $z_{25} := 1$ der

Operationsteil des Eingabe-Halt-Befehls in eine Addition als Einzelbefehl umgestellt. Bei Eingabe eines Zeichens der anderen Klasse wird durch z_{29}: = 0 der Operationsteil in einen unbedingten Sprung umgestellt. In beiden Fällen ist die Umstellungsbedingung nur durch das einzugebende Zeichen bestimmt und hängt nicht vom Inhalt des Adreßregisters ab. Der Operationszusatz wird nicht verändert. Phase IV wird mit der umgestellten Operation ausgeführt.

Bei Eingabe im **R 300-Code** wird zusätzlich das einzugebende Zeichen auf Paarigkeit geprüft; bei Paarigkeit wird der Ablauf der Befehlsschleife unterbrochen. Die Zeichen „Irrung Zeichen" und „Transportloch im Vorspann" unterliegen einer Sonderbehandlung; sie werden von der Paritätsprüfung ausgeschlossen und nicht in das Adreßregister übernommen.

Das Zeichen „Transportloch" außerhalb des Vorspannes auf dem Lochband unterliegt der Paritätsprüfung.

Bei Tastatur- und Schreibwerkeingabe entfallen diese Sonderzeichen. Mit dem Niederdrücken einer Taste finden Paritätsprüfung und Auswertung der Operationsumstellungsbedingung statt; beim Loslassen der Taste beginnt die Fortsetzung der Befehlsschleife mit der disjunktiven Übernahme des Zeichens ins Adreßregister. (Die Lochbandeingabe im R 300-Code ist in Abb. 16 beschrieben).

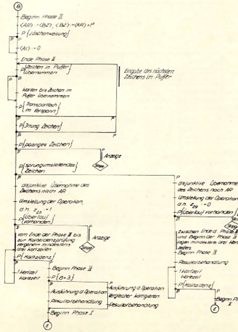


Abb. 16

Die Bedingung für die Umstellung des Operationsteils in einen unbedingten Sprung ist durch den logischen Ausdruck

$(1 \cdot 2 \cdot 4 \cdot 6) \vee (\bar{1} \cdot 2 \cdot 4 \cdot \bar{7}) \vee (1 \cdot 2 \cdot 4 \cdot 7) \vee (\bar{1} \cdot \bar{2} \cdot \bar{3} \cdot \bar{4} \cdot 7) \vee (1 \cdot \bar{2} \cdot 3 \cdot 4) \vee (2 \cdot 4 \cdot 6 \cdot 8)$

gegeben, wobei die einzelnen Argumente den Wert 1 haben, wenn der betreffende Kanal gelocht ist. In Sprung wird umgestellt, wenn der Ausdruck den Wert 1 besitzt. Im einzelnen ist die Umstellung der Operation aus Anlage 1 zu entnehmen. Die Lochkanäle 5 (Prüfkanal) und 8 haben für die Einstellung des Adreßregisters keine Bedeutung; der Code wird so verdichtet, daß die disjunktive Verknüpfung in den sechs niedrigsten Bits des Adreßregisters stattfindet (Abb. 17).

Lochkanal	7	6	4	3	2	1
Wertigkeit des Bit im einzugebenden Zeichen	2^5	2^4	2^3	2^2	2^1	2^0
Adreßregister	z_{18}	z_{19}	z_{20}	z_{21}	z_{22}	z_{23}

Abb. 17

Lochkanal	1	2	3	4	5
Wertigkeit des Bit im einzugebenden Zeichen	2^4	2^3	2^2	2^1	2^0
Adreßregister	z_{19}	z_{20}	z_{21}	z_{22}	z_{23}

Abb. 18

Beispiele:

Der Eingabe-Halt-Befehl
100 011
wird nach Eingabe von „," in
173 001
und nach Eingabe von „9" in
111 211
umgestellt.

Bei Eingabe im **Telegrafenalphabet Nr. 2** entfallen Paritätsprüfung und Zeichen mit Sonderbehandlung. Schreibwerkeingabe ist nur im R 300-Code möglich.

Die Bedingung für die Umstellung des Operationsteils in einen unbedingten Sprung ist durch

$(3 \cdot \bar{4} \cdot \bar{5}) \vee (\bar{1} \cdot \bar{2} \cdot \bar{3} \cdot 4 \cdot 5) \vee (\bar{1} \cdot 2 \cdot 3 \cdot 4 \cdot \bar{5})$

gegeben. Im einzelnen ist die Umstellung der Operation aus Anlage 2 zu entnehmen. Die disjunktive Verknüpfung findet in den fünf Bits der Sektornummer im Adreßregister statt (Abb. 18).

Beispiele:

Der Eingabe-Halt-Befehl
110 011
wird nach Eingabe von „," in
114 001
und nach Eingabe von „9" in
130 211
umgestellt.

4.3.2.3. **Unbedingter Stop 1. Art**

Der Befehl ist durch A = 2 und Z < 4 charakterisiert. Der Ablauf der Befehlsschleife wird vor der Wartezeit W_2 unterbrochen.

Durch die Starttaste kann die Abarbeitung der Befehlsschleife fortgesetzt werden. Dabei wird zunächst der Inhalt des Operationsregisters disjunktiv mit der Operation 240 verknüpft, so daß im Operationsregister eine Operation der Form 26Z, d. h. eine Konjunktion mit negiertem Akkumulatorinhalt steht. Die Fortsetzung verläuft wie bei dieser Operation.

Durch die Tastenkombination H Ü R wird der Inhalt des Operationsregisters in derselben Weise wie durch die Start-Taste verändert; außerdem wird im Adreßregister die Adresse 7777 eingestellt. Phase IV wird also mit einem Befehl 7777 26 Z ausgeführt.

Die Tastenkombination H M R verknüpft den Inhalt des Operationsregisters disjunktiv mit der Operation 025, so daß 025 oder 027 entsteht. Da Phase III bereits abgelaufen ist, wird von dieser Operation nur die Resultatbehandlung ausgeführt und anschließend der nächste Befehl aus der im Adreßteil des Stop-Befehls angegebenen Speicherzelle entnommen (s. Sprung mit Rückkehrabsicht). (Abb. 19)

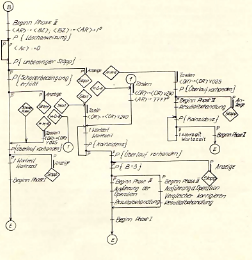


Abb. 19

4.3.2.4. **Sprung mit Rückkehrabsicht**

Der Befehl ist durch A = 2 und Z ≧ 4 charakterisiert. Dabei laufen die Phasen I, II normal ab. In Phase III wird zusätzlich der (eventuell gelöschte) Akkumulatorinhalt mit dem (inzwischen durch die Phasen I und III um eine Adreßeinheit erhöhten)

Befehlszählerinhalt disjunktiv verknüpft; das Ergebnis der Disjunktion steht im Akkumulator. Wegen $Z \geqq 4$ ist die in Phase IV auszuführende Resultatbehandlung eine Linksverschiebung oder eine zyklische Linksverschiebung. Schließlich wird der Sprung nach der im Adreßteil angegebenen Speicherzelle ausgeführt. (Abb. 20)

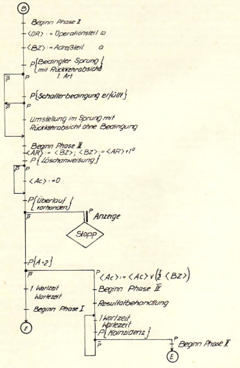


Abb. 20

Die Operationsteile 025, 027 haben dieselbe Wirkung; weil im Ergebnis der Disjunktion in Phase III hier $z_0 = 0$ ist, sind Linksverschiebung und zyklische Linksverschiebung nicht voneinander verschieden. Nach dieser Operation steht im Akkumulator ein unbedingter Sprung, dessen Adreßteil der um eine Adresseneinheit erhöhte Befehlszählerinhalt ist; wenn die Zieladresse des Befehls der erste Befehl eines Unterprogrammes ist, steht im Akkumulator also der Rücksprungbefehl ins Hauptprogramm.

4.3.2.5. **Ausgabe-Halt**

Der Befehl ist durch $A = 3$ charakterisiert. Sein Ablauf hängt davon ab, ob das höchste Adreßbit z_{12} mit 0 oder 1 besetzt ist. Ist $z_{12} = 1$, so bewirkt der Befehl die Anwahl eines Eingabe-Ausgabe-Kanals; falls $z_{12} = 0$ ist, so wird über das angewählte Ausgabegerät ein durch den Adreßteil bestimmtes Zeichen ausgegeben. In beiden Fällen wird der Akkumulatorinhalt nur der Löschanweisung unterworfen. Die Umstellung der Operation nach Phase III in Überlaufsprung ist für die Wirkung des Befehls bedeutungslos, weil die Auswertung der Sprungbedingung nur vorher in Phase II erfolgen kann. Da Phase IV nicht abläuft, wird keine Resultatbehandlung ausgeführt (s. Überlaufsprung).

Der Rechner besitzt drei Eingabe-Ausgabe-Kanäle (E-A-Kanäle), die von 0 bis 2 numeriert sind. Die Zuordnung von peripheren Geräten zu den Kanälen und die Anwahladressen sind im einzelnen folgende:

E-A-Kanal	Anwahl-adresse	periphere Geräte	Bemerkungen
0	4700	Tastatur Schreibwerk-eingabe	Die Unterscheidung zwischen Tastatur und Schreibwerk erfolgt durch einen Schalter am Bedienungs-pult (s. Bedienungsanleitung)
1	4600	Leser 1 Stanzer	Bei folgendem Eingabe-Halt arbeitet Leser 1, bei Ausgabe-Halt der Stanzer
2	4500	Leser 2 Schreibwerk-ausgabe	Bei folgendem Eingabe-Halt arbeitet Leser 2, bei Ausgabe-Halt das Schreibwerk

Jede Anwahl bleibt bei eingeschaltetem Rechner solange erhalten, bis sie durch eine neue Anwahl gelöscht wird. Wird versehentlich eine nicht existierende Anwahl programmiert, so stoppt der Rechner mit Anzeige von „A" (s. Bedienungsanleitung).

Die Ausgabe erfolgt 1-Zeichen-gepuffert. Unter der Voraussetzungen, daß ein Ausgabegerät angewählt ist, wird ein Zeichen in den 1-Zeichen-Puffer übertragen, sobald dieser frei ist. Während aus dem Puffer die Ausgabe des Zeichens über das angewählte Ausgabegerät erfolgt, wird gleichzeitig die Abarbeitung der Befehlsschleife fortgesetzt. (Abb. 21)

Bei Ausgabe über das Schreibwerk wird das auszugebende Zeichen durch die sechs Bits z_{14} bis z_{15} und z_{17} bis z_{20} des Adreßteils nach dem in der Schreibmaschine verdrahteten Code bestimmt; die Bits z_{13}, z_{16} werden nicht ausgewertet. Nur die auf der Schreibwerktastatur vorhandenen Zeichen können gedruckt werden. (s. Anhang 1)

Die meisten zulässigen Ausgabeadressen sind doppelt mit Zeichen belegt; die Unterscheidung erfolgt mittels der Umschaltzeichen KB und GB.

Die Leistung des Schreibwerks von 10 Anschlägen pro Sekunde wird voll ausgenutzt, wenn zwischen zwei aufeinanderfolgenden Ausgabe-Halt-Befehlen nicht mehr als 28 Trommelumdrehungen liegen.

Bei Ausgabe über den Lochbandstanzer ist die zu stanzende Bitkombination mit den Bits z_{13} bis z_{20} identisch; mit 1 besetzte Bits werden gelocht, mit 0 besetzte bleiben ungelocht. Damit kann jeder beliebige Code mit max. 8 Kanälen realisiert werden. Die Zuordnung zwischen den einzelnen Lochkanälen und den Bits des Adreßregisters ist dabei die folgende:

Lochkanal	Bit
1	z_{20}
2	z_{19}
3	z_{18}
4	z_{17}
5	z_{16}
6	z_{15}
7	z_{14}
8	z_{13}

Bei voller Leistung des Stanzers von 50 Zeichen pro Sekunde können zwischen zwei aufeinanderfolgenden Ausgabe-Halt-Befehlen maximal 4,5 Trommelumdrehungen liegen.

Die acht formal verschiedenen Operationsteile unterscheiden sich in ihrer Wirkung nur durch vorhandene oder fehlende Löschanweisung.

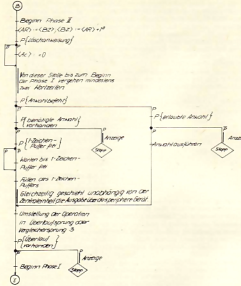


Abb. 21

4.3.2.6. **Negativsprung**

Der Befehl ist durch A = 4 charakterisiert. Sein Ablauf hängt vom Vorzeichen des Akkumulatorinhaltes ab; maßgebend ist das Vorzeichen während Phase II.

Unter der Bedingung $z_0 = 1$ wird er wie ein unbedingter Sprung ausgeführt. Andernfalls wird nur die Löschanweisung ausgeführt; anschließend wird der nächste Befehl in natürlicher Reihenfolge abgearbeitet. (Abb. 15)

Die Operationsteile 041, 043, 045, 047 haben dieselbe Wirkung.

4.3.2.7. **Positivsprung**

Der Befehl ist durch A = 5 charakterisiert. Sein Ablauf hängt vom Vorzeichen des Akkumulatorinhaltes ab; maßgebend ist das Vorzeichen während Phase II.

Unter der Bedingung $z_0 = 0$ wird er wie ein unbedingter Sprung ausgeführt. Andernfalls wird nur die Löschanweisung ausgeführt; anschließend wird der nächste Befehl in natürlicher Reihenfolge abgearbeitet. (Abb. 15)

Die Operationsteile 051, 053, 055, 057 haben dieselbe Wirkung. Da die Resultatbehandlung nur bei $z_0 = 0$ ausgeführt wird, liefern Linksverschiebung und zyklische Linksverschiebung, also auch die Operationen 054, 056 dasselbe Resultat.

4.3.2.8. **Bedingter Stop 1. Art**

Der Befehl ist durch A = 6 charakterisiert. Sein Ablauf hängt von einer gewissen Schalterbedingung ab (s. Bedienungsanleitung).

Bei erfüllter Schalterbedingung ist der Ablauf bis zur Unterbrechung der Befehlsschleife wie für A = 2. Die Starttaste sowie die Tastenkombination H Ü R bewirken ebenfalls dieselbe Fortsetzung wie für A = 2.

Durch die Tastenkombination H M R wird durch disjunktive Verknüpfung wie für A = 2 im Operationsregister ein bedingter Sprung mit Rückkehrabsicht 065 oder 067 eingestellt.

Da die Auswertung der Sprungbedingung nur in Phase II stattfinden kann, wird Phase IV nicht durchlaufen, sondern es schließt sich Phase I an. (s. bedingter Sprung mit Rückkehrabsicht).

4.3.2.9. **Bedingter Sprung mit Rückkehrabsicht**

Der Befehl ist durch A = 6 und $Z \geq 4$ charakterisiert. Sein Ablauf hängt von einer gewissen Schalterbedingung ab (s. Bedienungsanleitung).

Bei erfüllter Schalterbedingung ist der Ablauf der gleiche wie für A = 2; dann wirkt der Befehl wie ein (unbedingter) Sprung mit Rückkehrabsicht. Anderenfalls vergeht anstelle der Phase IV eine ungenutzte Wortzeit, an die sich Phase I anschließt; der Befehl wirkt bis auf die eventuelle Löschung des Akkumuatorinhaltes als Leerbefehl. (Abb. 20)

Die Operationsteile 065, 067 haben dieselbe Wirkung. Weil im Ergebnis der Disjunktion in Phase III bei erfüllter Schalterbedingung $z_0 = 0$ ist, sind Linksverschiebung und zyklische Linksverschiebung voneinander nicht verschieden.

4.3.2.10. **Überlaufsprung**

Der Befehl ist durch A = 7 charakterisiert. Sein Ablauf hängt davon ab, ob ein Überlauf vorhanden ist oder nicht.

Überlauf wird nur bei Addition, Subtraktion, bedingter Addition oder R-Operation als Einzelbefehl oder markierter Einzelbefehl ohne Resultatbehandlung registriert. Als Überlauf gilt Verschiedenheit des Übertrages von z_1 nach z_0 vom Übertrag von z_0 zur nicht mehr vorhandenen Maschinenstelle, d. h. Überschreiten des Zahlenbereiches. Wenn ein Überlauf vorhanden ist, wird der Befehl wie ein unbedingter Sprung ausgeführt; gleichzeitig wird der Überlauf gelöscht. Anderenfalls wird nur eine eventuelle Löschung des Akkumulatorinhaltes ausgeführt; anschließend wird der nächste Befehl in natürlicher Reihenfolge abgearbeitet. (Abb. 22)

Die Operationsteile 071, 073, 075, 077 haben dieselbe Wirkung.

4.3.3. **Organisationsbefehle 2. Art**

Organisationsbefehle 2. Art sind durch B = 1 charakterisiert.

4.3.3.1. **Unbedingter Sprung**

Der Befehl ist durch A = 0 charakterisiert. Er hat die gleiche Wirkung wie der unbedingte Sprung 1. Art. (Abb. 15)

4.3.3.2. **Eingabe-Halt 2. Art**

Der Befehl ist durch A = 1 charakterisiert. Sein Ablauf ist dem Eingabe-Halt 1. Art analog. Wegen B = 1 entsteht bei Umstellung des Operationsteils in unbedingtem Sprung eine Operation der Form 10 Z, bei Umstellung in Addition eine Operation der Form 31 Z. Da die vorbereitende Setzung des Vergleichers 3 bei markierten Einzelbefehlen an einer früheren Stelle der Befehlsschleife erfolgt, wird im letzten Fall der Vergleichszustand nur korrigiert. Wenn beim Vergleich die Gleichheit richtig ist, wird keine Korrektur ausgeführt, sondern der zufällige Vergleichszustand bleibt erhalten; anderenfalls wird durch die Korrektur der richtige Vergleicher gesetzt. Im übrigen wird Phase IV der Operation 31 Z normal ausgeführt. (Lochbandeingabe s. Abb. 16)

4.3.3.3. **Unbedingter Stop 2. Art**

Der Befehl ist durch A = 2 und $Z < 4$ charakterisiert. Sein Ablauf stimmt bis zur Unterbrechung der Befehlsschleife mit dem unbedingten Stop 1. Art überein.

Durch die Starttaste wird als Ergebnis der disjunktiven Verknüpfung im Operationsregister eine Operation der Form 36 Z eingestellt. Da die vorbereitende Setzung des Vergleichers 3 bei markierten Einzelbefehlen an einer früheren Stelle der Befehlsschleife erfolgt, wird jetzt der Vergleichszustand nur korrigiert. Wenn beim Vergleich die Gleichheit richtig ist, wird keine Korrektur durchgeführt, sondern der zufällig

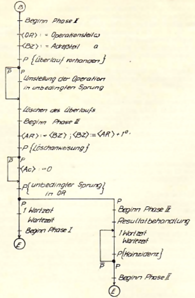


Abb. 22

vorhandene Vergleichszustand bleibt erhalten; anderenfalls wird durch die Korrektur der richtige Vergleicher gesetzt. Im übrigen wird Phase IV der Operation 36 Z normal ausgeführt.

Durch die Tastenkombination H Ü R wird der Inhalt des Operationsregisters in derselben Weise wie durch die Starttaste verändert; außerdem wird im Adreßregister die Adresse 7777 eingestellt. Phase IV wird also mit einem Befehl 7777 32 Z ausgeführt.

Die Tastenkombination H M R verknüpft den Inhalt des Operationsregisters disjunktiv mit der Operation 025, so daß 125 oder 127 entsteht. Da Phase III bereits abgelaufen ist, wird von dieser Operation nur die Resultatbehandlung ausgeführt und anschließend der nächste Befehl aus der im Adreßteil des Stop-Befehls angegebenen Speicherzelle entnommen (s. Sprung mit Rückkehrabsicht 2. Art). (Abb. 19)

4.3.3.4. Sprung mit Rückkehrabsicht

Der Befehl ist durch $A = 2$ und $Z \geq 4$ charakterisiert. Er hat dieselbe Wirkung wie der Sprung mit Rückkehrabsicht 1. Art. (Abb. 20)

4.3.3.5. Ausgabe-Halt

Der Befehl ist durch $A = 3$ charakterisiert. Er unterscheidet sich in seiner Wirkung nicht vom Ausgabe-Halt 1. Art. Im inneren Ablauf der Befehlsschleife wird nach

Phase III die Operation in Vergleichersprung 3 umgestellt (s. Vergleichersprung 3), was bedeutungslos ist, da die Auswertung der Sprungbedingung nur in Phase II erfolgen kann. (Abb. 21)

4.3.3.6. Vergleichersprung 1

Der Befehl ist durch A = 4 charakterisiert. Sein Ablauf hängt vom vorhandenen Vergleichszustand ab.

Ist einer der Vergleicher 1 oder 3 gesetzt, wird der Befehl wie ein unbedingter Sprung ausgeführt. Anderenfalls wird nur die Löschanweisung ausgeführt; anschließend wird der nächste Befehl in natürlicher Reihenfolge abgearbeitet. (Abb. 15)

Die Operationsteile 141, 143, 145, 147 haben dieselbe Wirkung.

4.3.3.7. Vergleichersprung 2

Der Befehl ist durch A = 5 charakterisiert. Sein Ablauf hängt vom vorhandenen Vergleichszustand ab.

Ist einer der Vergleicher 2 oder 3 gesetzt, wird der Befehl wie ein unbedingter Sprung ausgeführt. Anderenfalls wird nur die Löschanweisung ausgeführt; anschließend wird der nächste Befehl in natürlicher Reihenfolge abgearbeitet. (Abb. 15)

Die Operationsteile 151, 153, 155, 157 haben dieselbe Wirkung.

4.3.3.8. Bedingter Stop 2. Art

Der Befehl ist durch A = 6 und Z < 4 charakterisiert. Sein Ablauf hängt von derselben Schalterbedingung wie beim unbedingten Stop 1. Art und gleicht diesem bei erfüllter Schalterbedingung bis zur Unterbrechung der Befehlsschleife sowie bei nichterfüllter Schalterbedingung.

Bei erfüllter Schalterbedingung ist die Wirkung der Tastenkombination H M R dieselbe wie beim bedingten Stop 1. Art; es wird die Operation 165 oder 167 eingestellt. Durch die Start-Taste wird eine Operation der Gestalt 36 Z eingestellt. Die Tastenkombination H Ü R stellt als Ergebnis der disjunktiven Verknüpfung einen Befehl der Form 7777 36 Z ein.

Die beiden letzten Tastenkombinationen unterscheiden sich also nur durch das zusätzliche Vergleichersetzen vom bedingten Stop 1. Art.

4.3.3.9. Leer-Befehl

Der Befehl ist durch A = 6 und Z ≧ 4 charakterisiert. Er stellt formal einen bedingten Sprung mit Rückkehrabsicht 2. Art dar und wirkt unabhängig von der Schalterstellung wie ein unbedingter Sprung mit nicht erfüllter Bedingung. (Abb. 20)

Die Operationsteile 165, 167 bewirken die Löschung des Akkumulatorinhaltes; die Operationsteile 164, 166 führen ohne Wirkung zur Abarbeitung des nächsten Befehls in natürlicher Reihenfolge.

4.3.3.10. Vergleichersprung 3

Der Befehl ist durch A = 7 charakterisiert. Sein Ablauf hängt vom vorhandenen Vergleichszustand ab.

Ist der Vergleicher 3 gesetzt, wird der Befehl wie ein unbedingter Sprung ausgeführt. Andernfalls wird nur die Löschanweisung ausgeführt; anschließend wird der nächste Befehl in natürlicher Reihenfolge abgearbeitet. (Abb. 15)

Die Operationsteile 171, 173, 175, 177 haben dieselbe Wirkung.

4.4. Liste der Operationen

Im folgenden sind alle Operationsteile in Gruppen mit verschiedener Wirkung zusammengefaßt. Die Reihenfolge ist lexikographisch. Unterschiedlicher Zeitbedarf wurde nicht berücksichtigt.

1. **000**, 100
   unbedingter Sprung ohne Löschen, ohne Resultatbehandlung
2. **001**, 003, 005, 007, 101, 103, 105, 107
   unbedingter Sprung mit Löschen

3. **002**, 102
unbedingter Sprung mit Rechtsverschiebung

4. **004**, 104
unbedingter Sprung mit Linksverschiebung

5. **006**, 106
unbedingter Sprung mit zyklischer Linksverschiebung

6. **010**
Eingabe-Halt 1. Art ohne Löschen, ohne Resultatbehandlung

7. **011**
Eingabe-Halt 1. Art mit Löschen, ohne Resultatbehandlung

8. **012**
Eingabe-Halt 1. Art ohne Löschen, mit Rechtsverschiebung

9. **013**
Eingabe-Halt 1. Art mit Löschen, mit Rechtsverschiebung

10. **014**
Eingabe-Halt 1. Art ohne Löschen, mit Linksverschiebung

11. **015**
Eingabe-Halt 1. Art mit Löschen, mit Linksverschiebung

12. **016**
Eingabe-Halt 1. Art ohne Löschen, mit zyklischer Linksverschiebung

13. **017**
Eingabe-Halt 1. Art mit Löschen, mit zyklischer Linksverschiebung

14. **020**
unbedingter Stop 1. Art ohne Löschen, ohne Resultatbehandlung

15. **021**, 023
unbedingter Stop 1. Art mit Löschen

16. **022**
unbedingter Stop 1. Art ohne Löschen, mit Rechtsverschiebung

17. **024**, 124
Sprung mit Rückkehrabsicht ohne Löschen, mit Linksverschiebung

18. **025**, 027, 125, 127
Sprung mit Rückkehrabsicht mit Löschen, mit (zyklischer oder azyklischer) Linksverschiebung

19. **026**, 126
Sprung mit Rückkehrabsicht ohne Löschen, mit zyklischer Linksverschiebung

20. **030**, 032, 034, 036, 130, 132, 134, 136
Ausgabe-Halt ohne Löschen

21. **031**, 033, 035, 037, 131, 133, 135, 137
Ausgabe-Halt mit Löschen

22. **040**
Negativsprung ohne Löschen, ohne Resultatbehandlung

23. **041**, 043, 045, 047
Negativsprung mit Löschen

24. **042**
Negativsprung ohne Löschen, mit Rechtsverschiebung

25. **044**
Negativsprung ohne Löschen, mit Linksverschiebung

26. **046**
Negativsprung ohne Löschen, mit zyklischer Linksverschiebung

27. **050**
Positivsprung ohne Löschen, ohne Resultatbehandlung

28. **051**, 053, 055, 057
Positivsprung mit Löschen

29. **052**
Positivsprung ohne Löschen, mit Rechtsverschiebung

30. **054**, 056
Positivsprung ohne Löschen, mit (zyklischer oder azyklischer) Linksverschiebung

31. **060**
bedingter Stop 1. Art ohne Löschen, ohne Resultatbehandlung

32. **061**, 063
bedingter Stop 1. Art mit Löschen
33. **062**
bedingter Stop 1. Art ohne Löschen, mit Rechtsverschiebung
34. **064**
bedingter Sprung mit Rückkehrabsicht ohne Löschen, mit Linksverschiebung
35. **065**, 067
bedingter Sprung mit Rückkehrabsicht mit Löschen, mit (zyklischer oder azyklischer) Linksverschiebung
36. **066**
bedingter Sprung mit Rückkehrabsicht ohne Löschen, mit zyklischer Linksverschiebung
37. **070**
Überlaufsprung ohne Löschen, ohne Resultatbehandlung
38. **071**, 073, 075, 077
Überlaufsprung mit Löschen
39. **072**
Überlaufsprung ohne Löschen, mit Rechtsverschiebung
40. **074**
Überlaufsprung ohne Löschen, mit Linksverschiebung
41. **076**
Überlaufsprung ohne Löschen, mit zyklischer Linksverschiebung
42. **110**
Eingabe-Halt 2. Art ohne Löschen, ohne Resultatbehandlung
43. **111**
Eingabe-Halt 2. Art mit Löschen, ohne Resultatbehandlung
44. **112**
Eingabe-Halt 2. Art ohne Löschen, mit Rechtsverschiebung
45. **113**
Eingabe-Halt 2. Art mit Löschen, mit Rechtsverschiebung
46. **114**
Eingabe-Halt 2. Art ohne Löschen, mit Linksverschiebung
47. **115**
Eingabe-Halt 2. Art mit Löschen, mit Linksverschiebung
48. **116**
Eingabe-Halt 2. Art ohne Löschen, mit zyklischer Linksverschiebung
49. **117**
Eingabe-Halt 2. Art mit Löschen, mit zyklischer Linksverschiebung
50. **120**
unbedingter Stop 2. Art ohne Löschen, ohne Resultatbehandlung
51. **121**, 123
unbedingter Stop 2. Art mit Löschen
52. **122**
unbedingter Stop 2. Art ohne Löschen, mit Rechtsverschiebung
53. **140**
Vergleichersprung 1 ohne Löschen, ohne Resultatbehandlung
54. **141**, 143, 145, 147
Vergleichersprung 1 mit Löschen
55. **142**
Vergleichersprung 1 ohne Löschen, mit Rechtsverschiebung
56. **144**
Vergleichersprung 1 ohne Löschen, mit Linksverschiebung
57. **146**
Vergleichersprung 1 ohne Löschen, mit zyklischer Linksverschiebung
58. **150**
Vergleichersprung 2 ohne Löschen, ohne Resultatbehandlung
59. **151**, 153, 155, 157
Vergleichersprung 2 mit Löschen
60. **152**
Vergleichersprung 2 ohne Löschen, mit Rechtsverschiebung
61. **154**
Vergleichersprung 2 ohne Löschen, mit Linksverschiebung
62. **156**
Vergleichersprung 2 ohne Löschen, ohne Resultatbehandlung
63. **160**
bedingter Stop 2. Art ohne Löschen, ohne Resultatbehandlung

64. **161**, 163
bedingter Stop 2. Art mit Löschen

65. **162**
bedingter Stop 2. Art ohne Löschen, mit Rechtsverschiebung

66. **164**, 166
Leer-Operation ohne Löschen (ohne Resultatbehandlung)

67. **165**, 167
201, 203, 205, 207, 233, 261, 263, 265, 267, 273
401, 403, 405, 407, 433, 461, 463, 465, 467, 473
501, 503, 505, 507, 533, 561, 563, 565, 567, 573
601, 603, 605, 607, 633, 661, 663, 665, 667, 673
701, 703, 705, 707, 733, 761, 763, 765, 767, 773
Leer-Operation mit Löschen

68. **170**
Vergleichersprung 3 ohne Löschen, ohne Resultatbehandlung

69. **171**, 173, 175, 177
Vergleichersprung 3 mit Löschen

70. **172**
Vergleichersprung 3 ohne Löschen, mit Rechtsverschiebung

71. **174**
Vergleichersprung 3 ohne Löschen, mit Linksverschiebung

72. **176**
Vergleichersprung 3 ohne Löschen, mit zyklischer Linksverschiebung

73. **200**
Einzel-Konjunktion im Akkumulator ohne Löschen, ohne Resultatbehandlung

74. **202**
Einzel-Konjunktion im Akkumulator ohne Löschen, mit Rechtsverschiebung

75. **204**
Einzel-Konjunktion im Akkumulator ohne Löschen, mit Linksverschiebung

76. **206**
Einzel-Konjunktion im Akkumulator ohne Löschen, mit zyklischer Linksverschiebung

77. **210**
Einzel-Addition ohne Löschen, ohne Resultatbehandlung

78. **211**
Einzel-Addition mit Löschen, ohne Resultatbehandlung, d. h. Lesebefehl

79. **212**
Einzel-Addition ohne Löschen, mit Rechtsverschiebung

80. **213**
Einzel-Addition mit Löschen, mit Rechtsverschiebung

81. **214**
Einzel-Addition ohne Löschen, mit Linksverschiebung

82. **215**
Einzel-Addition mit Löschen, mit Linksverschiebung

83. **216**
Einzel-Addition ohne Löschen, mit zyklischer Linksverschiebung

84. **217**
Einzel-Addition mit Löschen, mit zyklischer Linksverschiebung

85. **220**
Einzel-Subtraktion ohne Löschen, ohne Resultatbehandlung

86. **221**, 231
Einzel-Subtraktion mit Löschen, ohne Resultatbehandlung

87. **222**
Einzel-Subtraktion ohne Löschen, mit Rechtsverschiebung

88. **223**
Einzel-Subtraktion mit Löschen, mit Rechtsverschiebung

89. **224**
Einzel-Subtraktion ohne Löschen, mit Linksverschiebung

90. **225**, 235
Einzel-Subtraktion mit Löschen, mit Linksverschiebung

91. **226**
Einzel-Subtraktion ohne Löschen, mit zyklischer Linksverschiebung

92. **227**, 237
Einzel-Subtraktion mit Löschen, mit zyklischer Linksverschiebung

93. **230**
Einzelbefehl bedingte Addition ohne Löschen, ohne Resultatbehandlung

94. **232**
Einzelbefehl bedingte Addition ohne Löschen, mit Rechtsverschiebung
95. **234**
Einzelbefehl bedingte Addition ohne Löschen, mit Linksverschiebung
96. **236**
Einzelbefehl bedingte Addition ohne Löschen, mit zyklischer Linksverschiebung
97. **240**
Einzel-Transport ohne Löschen, ohne Resultatbehandlung
98. **241**, 243, 245, 247
Einzel-Transport mit Löschen, d. h. Löschbefehl
99. **242**
Einzel-Transport ohne Löschen, mit Rechtsverschiebung
100. **244**
Einzel-Transport ohne Löschen, mit Linksverschiebung
101. **246**
Einzel-Transport ohne Löschen, mit zyklischer Linksverschiebung
102. **250**
Einzel-Konjunktion im Speicher (ohne Löschen), ohne Resultatbehandlung
103. **251**
Einzel-R-Operation (mit Löschen), ohne Resultatbehandlung
104. **252**
Einzel-Konjunktion im Speicher (ohne Löschen) mit Rechtsverschiebung
105. **253**
Einzel-R-Operation (mit Löschen), mit Rechtsverschiebung
106. **254**
Einzel-Konjunktion im Speicher (ohne Löschen), mit Linksverschiebung
107. **255**
Einzel-R-Operation (mit Löschen), mit Linksverschiebung
108. **256**
Einzel-Konjunktion im Speicher (ohne Löschen), mit zyklischer Linksverschiebung
109. **257**
Einzel-R-Operation (mit Löschen), mit zyklischer Linksverschiebung
110. **260**
Einzel-Konjunktion mit negiertem Akkumulatorinhalt ohne Löschen, ohne Resultatbehandlung
111. **262**
Einzel-Konjunktion mit negiertem Akkumulatorinhalt ohne Löschen, mit Rechtsverschiebung
112. **264**
Einzel-Konjunktion mit negiertem Akkumulatorinhalt ohne Löschen, mit Linksverschiebung
113. **266**
Einzel-Konjunktion mit negiertem Akkumulatorinhalt ohne Löschen, mit zyklischer Linksverschiebung
114. **270**
Einzelbefehl bedingte Speicherlöschung (ohne Löschen), ohne Resultatbehandlung
115. **271**
Einzel-R-Operation (mit Löschen), ohne Resultatbehandlung
116. **272**
Einzelbefehl bedingte Speicherlöschung (ohne Löschen), mit Rechtsverschiebung
117. **274**
Einzelbefehl bedingte Speicherlöschung (ohne Löschen), mit Linksverschiebung
118. **275**
Einzel-R-Operation (mit Löschen), mit Linksverschiebung
119. **276**
Einzelbefehl bedingte Speicherlöschung (ohne Löschen), mit zyklischer Linksverschiebung
120. **277**
Einzel-R-Operation (mit Löschen), mit zyklischer Linksverschiebung
121. **300**
markierte Einzel-Konjunktion im Akkumulator ohne Löschen, ohne Resultatbehandlung

122. **301,** 303, 305, 307, 331, 333, 373
markierte Leer-Operation mit Löschen, d. h. reines Vergleichersetzen mit Löschen

123. **302**
markierte Einzel-Konjunktion im Akkumulator ohne Löschen, mit Rechtsverschiebung

124. **304**
markierte Einzel-Konjunktion im Akkumulator ohne Löschen, mit Linksverschiebung

125. **306**
markierte Einzel-Konjunktion im Akkumulator ohne Löschen, mit zyklischer Linksverschiebung

126. **310**
markierte Einzel-Addition ohne Löschen, ohne Resultatbehandlung

127. **311**
markierte Einzel-Addition mit Löschen, ohne Resultatbehandlung, d. h. Lesebefehl mit Vergleichersetzen

128. **312**
markierte Einzel-Addition ohne Löschen, mit Rechtsverschiebung

129. **313**
markierte Einzel-Addition mit Löschen, mit Rechtsverschiebung

130. **314**
markierte Einzel-Addition ohne Löschen, mit Linksverschiebung

131. **315**
markierte Einzel-Addition mit Löschen, mit Linksverschiebung

132. **316**
markierte Einzel-Addition ohne Löschen, mit zyklischer Linksverschiebung

133. **317**
markierte Einzel-Addition mit Löschen, mit zyklischer Linksverschiebung

134. **320**
markierte Einzel-Subtraktion ohne Löschen, ohne Resultatbehandlung

135. **321**
markierte Einzel-Subtraktion mit Löschen, ohne Resultatbehandlung, d. h. Lesebefehl für das Komplement mit Vergleichersetzen

136. **322**
markierte Einzel-Subtraktion ohne Löschen, mit Rechtsverschiebung

137. **323**
markierte Einzel-Subtraktion mit Löschen, mit Rechtsverschiebung

138. **324**
markierte Einzel-Subtraktion ohne Löschen, mit Linksverschiebung

139. **325,** 335
markierte Einzel-Subtraktion mit Löschen, mit Linksverschiebung

140. **326**
markierte Einzel-Subtraktion ohne Löschen, mit zyklischer Linksverschiebung

141. **327,** 337
markierte Einzel-Subtraktion mit Löschen, mit zyklischer Linksverschiebung

142. **330**
reines Vergleichersetzen ohne Löschen

143. **332**
markierter Einzelbefehl bedingte Addition ohne Löschen, mit Rechtsverschiebung

144. **334**
markierter Einzelbefehl bedingte Addition ohne Löschen, mit Linksverschiebung

145. **336**
markierter Einzelbefehl bedingte Addition ohne Löschen, mit zyklischer Linksverschiebung

146. **340**
markierter Einzel-Transport ohne Löschen, ohne Resultatbehandlung

147. **341,** 343, 345, 347
markierter Einzel-Transport mit Löschen, d. h. Löschbefehl mit Setzen des Vergleichers 3

148. **342**
markierter Einzel-Transport ohne Löschen, mit Rechtsverschiebung

149. **344**
markierter Einzel-Transport ohne Löschen, mit Linksverschiebung

150. **346**
markierter Einzel-Transport ohne Löschen, mit zyklischer Linksverschiebung

151. **350**
markierte Einzel-Konjunktion im Speicher (ohne Löschen), ohne Resultatbehandlung

152. **351**
markierte Einzel-R-Operation (mit Löschen), ohne Resultatbehandlung

153. **352**
markierte Einzel-Konjunktion im Speicher (ohne Löschen), mit Rechtsverschiebung

154. **353**
markierte Einzel-R-Operation (mit Löschen), mit Rechtsverschiebung

155. **354**
markierte Einzel-Konjunktion im Speicher (ohne Löschen), mit Linksverschiebung

156. **355**
markierte Einzel-R-Operation (mit Löschen), mit Linksverschiebung

157. **356**
markierte Einzel-Konjunktion im Speicher (ohne Löschen), mit zyklischer Linksverschiebung

158. **357**
markierte Einzel-R-Operation (mit Löschen), mit zyklischer Linksverschiebung

159. **360**
markierte Einzel-Konjunktion mit negiertem Akkumulatorinhalt ohne Löschen, ohne Resultatbehandlung

160. **361**, 363, 365, 367
markierte Leer-Operation mit Löschen, d. h. Leer-Operation mit Löschen und mit Setzen des Vergleichers 3

161. **362**
markierte Einzel-Konjunktion mit negiertem Akkumulatorinhalt ohne Löschen, mit Rechtsverschiebung

162. **364**
markierte Einzel-Konjunktion mit negiertem Akkumulatorinhalt ohne Löschen, mit Linksverschiebung

163. **366**
markierte Einzel-Konjunktion mit negiertem Akkumulatorinhalt ohne Löschen, mit zyklischer Linksverschiebung

164. **370**
markierter Einzel-Befehl bedingte Speicherlöschung (ohne Löschen), ohne Resultatbehandlung

165. **371**
markierte Einzel-R-Operation (mit Löschen), ohne Resultatbehandlung

166. **372**
markierter Einzelbefehl bedingte Speicherlöschung (ohne Löschen), mit Rechtsverschiebung

167. **374**
markierter Einzelbefehl bedingte Speicherlöschung (ohne Löschen), mit Linksverschiebung

168. **375**
markierte Einzel-R-Operation (mit Löschen), mit Linksverschiebung

169. **376**
markierter Einzelbefehl bedingte Speicherlöschung (ohne Löschen), mit zyklischer Linksverschiebung

170. **377**
markierte Einzel-R-Operation (mit Löschen), mit zyklischer Linksverschiebung

171. **400**
G-Konjunktion im Akkumulator ohne Löschen, ohne Resultatbehandlung

172. **402**
G-Konjunktion im Akkumulator ohne Löschen, mit Rechtsverschiebung

173. **404**
G-Konjunktion im Akkumulator ohne Löschen, mit Linksverschiebung

174. **406**
G-Konjunktion im Akkumulator ohne Löschen, mit zyklischer Linksverschiebung

175. **410**
G-Addition ohne Löschen, ohne Resultatbehandlung
176. **411**
G-Addition mit Löschen, ohne Resultatbehandlung
177. **412**
G-Addition ohne Löschen, mit Rechtsverschiebung
178. **413**
G-Addition mit Löschen, mit Rechtsverschiebung
179. **414**
G-Addition ohne Löschen, mit Linksverschiebung
180. **415**
G-Addition mit Löschen, mit Linksverschiebung
181. **416**
G-Addition ohne Löschen, mit zyklischer Linksverschiebung
182. **417**
G-Addition mit Löschen, mit zyklischer Linksverschiebung
183. **420**
G-Subtraktion ohne Löschen, ohne Resultatbehandlung
184. **421**
G-Subtraktion mit Löschen, ohne Resultatbehandlung
185. **422**
G-Subtraktion ohne Löschen, mit Rechtsverschiebung
186. **423**
G-Subtraktion mit Löschen, mit Rechtsverschiebung
187. **424**
G-Subtraktion ohne Löschen, mit Linksverschiebung
188. **425**
G-Subtraktion mit Löschen, mit Linksverschiebung
189. **426**
G-Subtraktion ohne Löschen, mit zyklischer Linksverschiebung
190. **427**
G-Subtraktion mit Löschen, mit zyklischer Linksverschiebung
191. **430**
G-Befehl bedingte Addition ohne Löschen, ohne Resultatbehandlung
192. **431**
G-Befehl bedingte Addition mit Löschen, ohne Resultatbehandlung
193. **432**
G-Befehl bedingte Addition ohne Löschen, mit Rechtsverschiebung
194. **434**
G-Befehl bedingte Addition ohne Löschen, mit Linksverschiebung
195. **435**
G-Befehl bedingte Addition mit Löschen, mit Linksverschiebung
196. **436**
G-Befehl bedingte Addition ohne Löschen, mit zyklischer Linksverschiebung
197. **437**
G-Befehl bedingte Addition mit Löschen, mit zyklischer Linksverschiebung
198. **440**
G-Transport ohne Löschen, ohne Resultatbehandlung
199. **441**, 443, 445, 447
541, 543, 545, 547
G-Transport mit Löschen, d. h. G-Löschbefehl
200. **442**
G-Transport ohne Löschen, mit Rechtsverschiebung
201. **444**
G-Transport ohne Löschen, mit Linksverschiebung
202. **446**
G-Transport ohne Löschen, mit zyklischer Linksverschiebung
203. **450**
G-Konjunktion im Speicher (ohne Löschen), ohne Resultatbehandlung
204. **451**
G-R-Operation (mit Löschen), ohne Resultatbehandlung
205. **452**
G-Konjunktion im Speicher (ohne Löschen), mit Rechtsverschiebung
206. **453**
G-R-Operation (mit Löschen), mit Rechtsverschiebung

207. **454**
G-Konjunktion im Speicher (ohne Löschen), mit Linksverschiebung

208. **455**
G-R-Operation (mit Löschen), mit Linksverschiebung

209. **456**
G-Konjunktion im Speicher (ohne Löschen), mit zyklischer Linksverschiebung

210. **457**
G-R-Operation (mit Löschen), mit zyklischer Linksverschiebung

211. **460**
G-Konjunktion mit negiertem Akkumulatorinhalt ohne Löschen, ohne Resultatbehandlung

212. **462**
G-Konjunktion mit negiertem Akkumulatorinhalt ohne Löschen, mit Rechtsverschiebung

213. **464**
G-Konjunktion mit negiertem Akkumulatorinhalt ohne Löschen, mit Linksverschiebung

214. **466**
G-Konjunktion mit negiertem Akkumulatorinhalt ohne Löschen, mit zyklischer Linksverschiebung

215. **470**
G-Befehl bedingte Speicherlöschung (ohne Löschen), ohne Resultatbehandlung

216. **471**
G-R-Operation (mit Löschen), ohne Resultatbehandlung

217. **472**
G-Befehl bedingte Speicherlöschung (ohne Löschen), mit Rechtsverschiebung

218. **474**
G-Befehl bedingte Speicherlöschung (ohne Löschen), mit Linksverschiebung

219. **475**
G-R-Operation (mit Löschen), mit Linksverschiebung

220. **476**
G-Befehl bedingte Speicherlöschung (ohne Löschen), mit zyklischer Linksverschiebung

221. **477**
G-R-Operation (mit Löschen), mit zyklischer Linksverschiebung

222. **500**
GX-Konjunktion im Akkumulator ohne Löschen, ohne Resultatbehandlung

223. **502**
GX-Konjunktion im Akkumulator ohne Löschen, mit Rechtsverschiebung

224. **504**
GX-Konjunktion im Akkumulator ohne Löschen, mit Linksverschiebung

225. **506**
GX-Konjunktion im Akkumulator ohne Löschen, mit zyklischer Linksverschiebung

226. **510**
GX-Addition ohne Löschen, ohne Resultatbehandlung

227. **511**
GX-Addition mit Löschen, ohne Resultatbehandlung

228. **512**
GX-Addition ohne Löschen, mit Rechtsverschiebung

229. **513**
GX-Addition mit Löschen, mit Rechtsverschiebung

230. **514**
GX-Addition ohne Löschen, mit Linksverschiebung

231. **515**
GX-Addition mit Löschen, mit Linksverschiebung

232. **516**
GX-Addition ohne Löschen, mit zyklischer Linksverschiebung

233. **517**
GX-Addition mit Löschen, mit zyklischer Linksverschiebung

234. **520**
GX-Subtraktion ohne Löschen, ohne Resultatbehandlung

235. **521**
GX-Subtraktion mit Löschen, ohne Resultatbehandlung

236. **522**
GX-Subtraktion ohne Löschen, mit Rechtsverschiebung

237. **523**
GX-Subtraktion mit Löschen, mit Rechtsverschiebung

238. **524**
GX-Subtraktion ohne Löschen, mit Linksverschiebung

239. **525**
GX-Subtraktion mit Löschen, mit Linksverschiebung

240. **526**
GX-Subtraktion ohne Löschen, mit zyklischer Linksverschiebung

241. **527**
GX-Subtraktion mit Löschen, mit zyklischer Linksverschiebung

242. **530**
GX-Befehl bedingte Addition ohne Löschen, ohne Resultatbehandlung

243. **531**
GX-Befehl bedingte Addition mit Löschen, ohne Resultatbehandlung

244. **532**
GX-Befehl bedingte Addition ohne Löschen, mit Rechtsverschiebung

245. **534**
GX-Befehl bedingte Addition ohne Löschen, mit Linksverschiebung

246. **535**
GX-Befehl bedingte Addition mit Löschen, mit Linksverschiebung

247. **536**
GX-Befehl bedingte Addition ohne Löschen, mit zyklischer Linksverschiebung

248. **537**
GX-Befehl bedingte Addition mit Löschen, mit zyklischer Linksverschiebung

249. **540**
GX-Transport ohne Löschen, ohne Resultatbehandlung

250. **542**
GX-Transport ohne Löschen, mit Rechtsverschiebung

251. **544**
GX-Transport ohne Löschen, mit Linksverschiebung

252. **546**
GX-Transport ohne Löschen, mit zyklischer Linksverschiebung

253. **550**
GX-Konjunktion im Speicher (ohne Löschen), ohne Resultatbehandlung

254. **551**
GX-R-Operation (mit Löschen), ohne Resultatbehandlung

255. **552**
GX-Konjunktion im Speicher (ohne Löschen), mit Rechtsverschiebung

256. **553**
GX-R-Operation (mit Löschen), mit Rechtsverschiebung

257. **554**
GX-Konjunktion im Speicher (ohne Löschen), mit Linksverschiebung

258. **555**
GX-R-Operation (mit Löschen), mit Linksverschiebung

259. **556**
GX-Konjunktion im Speicher (ohne Löschen), mit zyklischer Linksverschiebung

260. **557**
GX-R-Operation (mit Löschen), mit zyklischer Linksverschiebung

261. **560**
GX-Konjunktion mit negiertem Akkumulatorinhalt ohne Löschen, ohne Resultatbehandlung

262. **562**
GX-Konjunktion mit negiertem Akkumulatorinhalt ohne Löschen, mit Rechtsverschiebung

263. **564**
GX-Konjunktion mit negiertem Akkumulatorinhalt ohne Löschen, mit Linksverschiebung

264. **566**
GX-Konjunktion mit negiertem Akkumulatorinhalt ohne Löschen, mit zyklischer Linksverschiebung

265. **570**
GX-Befehl bedingte Speicherlöschung (ohne Löschen), ohne Resultatbehandlung
266. **571**
GX-R-Operation (mit Löschen), ohne Resultatbehandlung
267. **572**
GX-Befehl bedingte Speicherlöschung (ohne Löschen), mit Rechtsverschiebung
268. **574**
GX-Befehl bedingte Speicherlöschung (ohne Löschen), mit Linksverschiebung
269. **575**
GX-R-Operation (mit Löschen), mit Linksverschiebung
270. **576**
GX-Befehle bedingte Speicherlöschung (ohne Löschen), mit zyklischer Linksverschiebung
271. **577**
GX-R-Operation (mit Löschen), mit zyklischer Linksverschiebung
272. **600**
W-Konjunktion im Akkumulator ohne Löschen, ohne Resultatbehandlung
273. **602**
W-Konjunktion im Akkumulator ohne Löschen, mit Rechtsverschiebung
274. **604**
W-Konjunktion im Akkumulator ohne Löschen, mit Linksverschiebung
275. **606**
W-Konjunktion im Akkumulator ohne Löschen, mit zyklischer Linksverschiebung
276. **610**
W-Addition ohne Löschen, ohne Resultatbehandlung
277. **611**
W-Addition mit Löschen, ohne Resultatbehandlung
278. **612**
W-Addition ohne Löschen, mit Rechtsverschiebung
279. **613**
W-Addition mit Löschen, mit Rechtsverschiebung
280. **614**
W-Addition ohne Löschen, mit Linksverschiebung
281. **615**
W-Addition mit Löschen, mit Linksverschiebung
282. **616**
W-Addition ohne Löschen, mit zyklischer Linksverschiebung
283. **617**
W-Addition mit Löschen, mit zyklischer Linksverschiebung
284. **620**
W-Subtraktion ohne Löschen, ohne Resultatbehandlung
285. **621**
W-Subtraktion mit Löschen, ohne Resultatbehandlung
286. **622**
W-Subtraktion ohne Löschen, mit Rechtsverschiebung
287. **623**
W-Subtraktion mit Löschen, mit Rechtsverschiebung
288. **624**
W-Subtraktion ohne Löschen, mit Linksverschiebung
289. **625**
W-Subtraktion mit Löschen, mit Linksverschiebung
290. **626**
W-Subtraktion ohne Löschen, mit zyklischer Linksverschiebung
291. **627**
W-Subtraktion mit Löschen, mit zyklischer Linksverschiebung
292. **630**
W-Befehl bedingte Addition ohne Löschen, ohne Resultatbehandlung
293. **631**
W-Befehl bedingte Addition mit Löschen, ohne Resultatbehandlung
294. **632**
W-Befehl bedingte Addition ohne Löschen, mit Rechtsverschiebung
295. **634**
W-Befehl bedingte Addition ohne Löschen, mit Linksverschiebung

296. **635**
W-Befehl bedingte Addition mit Löschen, mit Linksverschiebung

297. **636**
W-Befehl bedingte Addition ohne Löschen, mit zyklischer Linksverschiebung

298. **637**
W-Befehl bedingte Addition mit Löschen, mit zyklischer Linksverschiebung

299. **640**
W-Transport ohne Löschen, ohne Resultatbehandlung

300. **641**, 643, 645, 647
741, 743, 745, 747
W-Transport mit Löschen, d. h. W-Löschbefehl

301. **642**
W-Transport ohne Löschen, mit Rechtsverschiebung

302. **644**
W-Transport ohne Löschen, mit Linksverschiebung

303. **646**
W-Transport ohne Löschen, mit zyklischer Linksverschiebung

304. **650**
W-Konjunktion im Speicher (ohne Löschen), ohne Resultatbehandlung

305. **651**
W-R-Operation (mit Löschen), ohne Resultatbehandlung

306. **652**
W-Konjunktion im Speicher (ohne Löschen), mit Rechtsverschiebung

307. **653**
W-R-Operation (mit Löschen), mit Rechtsverschiebung

308. **654**
W-Konjunktion im Speicher (ohne Löschen), mit Linksverschiebung

309. **655**
W-R-Operation (mit Löschen), mit Linksverschiebung

310. **656**
W-Konjunktion im Speicher (ohne Löschen), mit zyklischer Linksverschiebung

311. **657**
W-R-Operation (mit Löschen), mit zyklischer Linksverschiebung

312. **660**
W-Konjunktion mit negiertem Akkumulatorinhalt ohne Löschen, ohne Resultatbehandlung

313. **662**
W-Konjunktion mit negiertem Akkumulatorinhalt ohne Löschen, mit Rechtsverschiebung

314. **664**
W-Konjunktion mit negiertem Akkumulatorinhalt ohne Löschen, mit Linksverschiebung

315. **666**
W-Konjunktion mit negiertem Akkumulatorinhalt ohne Löschen, mit zyklischer Linksverschiebung

316. **670**
W-Befehl bedingte Speicherlöschung (ohne Löschen), ohne Resultatbehandlung

317. **671**
W-R-Operation (mit Löschen), ohne Resultatbehandlung

318. **672**
W-Befehl bedingte Speicherlöschung (ohne Löschen), mit Rechtsverschiebung

319. **674**
W-Befehl bedingte Speicherlöschung (ohne Löschen), mit Linksverschiebung

320. **675**
W-R-Operation (mit Löschen), mit Linksverschiebung

321. **676**
W-Befehl bedingte Speicherlöschung (ohne Löschen), mit zyklischer Linksverschiebung

322. **677**
W-R-Operation (mit Löschen), mit zyklischer Linksverschiebung

323. **700**
WX-Konjunktion im Akkumulator ohne Löschen, ohne Resultatbehandlung

324. 702
WX-Konjunktion im Akkumulator ohne Löschen, mit Rechtsverschiebung

325. 704
WX-Konjunktion im Akkumulator ohne Löschen, mit Linksverschiebung

326. 706
WX-Konjunktion im Akkumulator ohne Löschen, mit zyklischer Linksverschiebung

327. 710
WX-Addition ohne Löschen, ohne Resultatbehandlung

328. 711
WX-Addition mit Löschen, ohne Resultatbehandlung

329. 712
WX-Addition ohne Löschen, mit Rechtsverschiebung

330. 713
WX-Addition mit Löschen, mit Rechtsverschiebung

331. 714
WX-Addition ohne Löschen, mit Linksverschiebung

332. 715
WX-Addition mit Löschen, mit Linksverschiebung

333. 716
WX-Addition ohne Löschen, mit zyklischer Linksverschiebung

334. 717
WX-Addition mit Löschen, mit zyklischer Linksverschiebung

335. 720
WX-Subtraktion ohne Löschen, ohne Resultatbehandlung

336. 721
WX-Subtraktion mit Löschen, ohne Resultatbehandlung

337. 722
WX-Subtraktion ohne Löschen, mit Rechtsverschiebung

338. 723
WX-Subtraktion mit Löschen, mit Rechtsverschiebung

339. 724
WX-Subtraktion ohne Löschen, mit Linksverschiebung

340. 725
WX-Subtraktion mit Löschen, mit Linksverschiebung

341. 726
WX-Subtraktion ohne Löschen, mit zyklischer Linksverschiebung

342. 727
WX-Subtraktion mit Löschen, mit zyklischer Linksverschiebung

343. 730
WX-Befehl bedingte Addition ohne Löschen, ohne Resultatbehandlung

344. 731
WX-Befehl bedingte Addition mit Löschen, ohne Resultatbehandlung

345. 732
WX-Befehl bedingte Addition ohne Löschen, mit Rechtsverschiebung

346. 734
WX-Befehl bedingte Addition ohne Löschen, mit Linksverschiebung

347. 735
WX-Befehl bedingte Addition mit Löschen, mit Linksverschiebung

348. 736
WX-Befehl bedingte Addition ohne Löschen, mit zyklischer Linksverschiebung

349. 737
WX-Befehl bedingte Addition mit Löschen, mit zyklischer Linksverschiebung

350. 740
WX-Transport ohne Löschen, ohne Resultatbehandlung

351. 742
WX-Transport ohne Löschen, mit Rechtsverschiebung

352. 744
WX-Transport ohne Löschen, mit Linksverschiebung

353. 746
WX-Transport ohne Löschen, mit zyklischer Linksverschiebung

354. 750
WX-Konjunktion im Speicher (ohne Löschen), ohne Resultatbehandlung

355. 751
WX-R-Operation (mit Löschen), ohne Resultatbehandlung

356. 752
WX-Konjunktion im Speicher (ohne Löschen), mit Rechtsverschiebung
357. 753
WX-R-Operation (mit Löschen), mit Rechtsverschiebung
358. 754
WX-Konjunktion im Speicher (ohne Löschen), mit Linksverschiebung
359. 755
WX-R-Operation (mit Löschen), mit Linksverschiebung
360. 756
WX-Konjunktion im Speicher (ohne Löschen), mit zyklischer Linksverschiebung
361. 757
WX-R-Operation (mit Löschen), mit zyklischer Linksverschiebung
362. 760
WX-Konjunktion mit negiertem Akkumulatorinhalt ohne Löschen, ohne Resultatbehandlung
363. 762
WX-Konjunktion mit negiertem Akkumulatorinhalt ohne Löschen, mit Rechtsverschiebung
364. 764
WX-Konjunktion mit negiertem Akkumulatorinhalt ohne Löschen, mit Linksverschiebung
365. 766
WX-Konjunktion mit negiertem Akkumulatorinhalt ohne Löschen, mit zyklischer Linksverschiebung
366. 770
WX-Befehl bedingte Speicherlöschung (ohne Löschen), ohne Resultatbehandlung
367. 771
WX-R-Operation (mit Löschen), ohne Resultatbehandlung
368. 772
WX-Befehl bedingte Speicherlöschung (ohne Löschen), mit Rechtsverschiebung
369. 774
WX-Befehl bedingte Speicherlöschung (ohne Löschen), mit Linksverschiebung
370. 775
WX-R-Operation (mit Löschen), mit Linksverschiebung
371. 776
WX-Befehl bedingte Speicherlöschung (ohne Löschen), mit zyklischer Linksverschiebung
372. 777
WX-R-Operation (mit Löschen), mit zyklischer Linksverschiebung

4.5. Zusammenstellung der häufigsten Operationen

Es werden diejenigen verdrahteten Operationen, die bei der praktischen Programmierung am häufigsten benötigt werden, angegeben.

Codierung	Bedeutung (für Adreßteil a)
200 (300)	Konjunktion im Akkumulator: $\langle Ac \rangle := \langle Ac \rangle \wedge \langle a \rangle$ (mit Vergleichersetzen)
211 (311)	Lesebefehl: $\langle Ac \rangle := \langle a \rangle$ (mit Vergleichersetzen)
210 (310)	Addition im Festkomma: $\langle Ac \rangle := \langle Ac \rangle + \langle a \rangle$ (mit Vergleichersetzen)
221 (321)	Lesebefehl für das Komplement: $\langle Ac \rangle := -\langle a \rangle$ (mit Vergleichersetzen)
220 (320)	Subtraktion im Festkomma $\langle Ac \rangle := \langle Ac \rangle - \langle a \rangle$ (mit Vergleichersetzen)

330	Vergleichersetzen
331	Vergleichersetzen: $\langle Ac \rangle := 0$
240 (340)	Speicherbefehl: $\langle a \rangle := \langle Ac \rangle$ (mit Vergleichersetzen)
241	Löschbefehl: $\langle Ac \rangle := 0$ $\langle a \rangle := 0$
250 (350)	Konjunktion im Speicher: $\langle a \rangle := \langle Ac \rangle \wedge \langle a \rangle$ (mit Vergleichersetzen)
251 (351)	R-Befehl: $\langle Ac \rangle := \langle \alpha \rangle$ (mit Vergleichersetzen)
271 (371)	R-Befehl: $\langle Ac \rangle := -\langle \alpha \rangle$ (mit Vergleichersetzen)
000	Unbedingter Sprung: $\rangle$ nächster Befehl $\langle := a$
040	Negativsprung: $\rangle$ nächster Befehl $\langle := \begin{cases} a, & \text{falls } \langle Ac \rangle < 0 \\ \langle BZ \rangle + 1^\circ, & \text{falls } \langle Ac \rangle \geqq 0 \end{cases}$
050	Positivsprung: $\rangle$ nächster Befehl $\langle := \begin{cases} a, & \text{falls } \langle AC \rangle \geqq 0 \\ \langle BZ \rangle + 1^\circ, & \text{falls } \langle AC \rangle < 0 \end{cases}$
070	Überlaufsprung: $\rangle$ nächster Befehl $\langle := \begin{cases} a, & \text{falls Überlauf} \\ \langle BZ \rangle + 1^\circ, & \text{falls kein Überlauf} \end{cases}$
140	Vergleichersprung 1 $\rangle$ nächster Befehl $\langle := \begin{cases} a, & \text{falls Vergleicher 1 oder 3} \\ \langle BZ \rangle + 1^\circ, & \text{falls weder Vergleicher 1 noch 3} \end{cases}$
150	Vergleichersprung 2 $\rangle$ nächster Befehl $\langle := \begin{cases} a, & \text{falls Vergleicher 2 oder 3} \\ \langle BZ \rangle + 1^\circ, & \text{falls weder Vergleicher 2 noch 3} \end{cases}$
170	Vergleichersprung 3 $\rangle$ nächster Befehl $\langle := \begin{cases} a, & \text{falls Vergleicher 3} \\ \langle BZ \rangle + 1^\circ, & \text{falls nicht Vergleicher 3} \end{cases}$
021	Unbedingter Stop: $\langle Ac \rangle := 0$
061	Bedingter Stop: $\langle Ac \rangle := 0$
025	Sprung mit Rückkehrabsicht: $\langle Ac \rangle := \langle BZ \rangle + 1^\circ$ $\rangle$ nächster Befehl $\langle := a$
030	Ausgabe-Halt Anwahl bzw. Ausgabe eines Zeichens

5. Programmierbeispiele

5.1. Vertauschung zweier Wörter

Gegeben sei $\langle 3000 \rangle = a$, $\langle 3001 \rangle = b$. Die beiden Zahlen a und b sollen vertauscht werden. Das Programm soll in Zelle 3003 beginnen.

Adr.	Befehl	Bemerkungen
3003	3000 211	$\langle Ac \rangle := a$
3004	3002 240	$\langle 3002 \rangle := a$
3005	3001 211	$\langle Ac \rangle := b$
3006	3000 240	$\langle 3000 \rangle := b$
3007	3002 211	$\langle Ac \rangle := a$
3010	3001 240	$\langle 3001 \rangle := a$

5.2. Algebraische Summe

Gegeben sei $\langle 3000 \rangle = a$, $\langle 3001 \rangle = b$, $\langle 3002 \rangle = c$. Es soll $x = a - b + c$ berechnet und nach 3003 gespeichert werden. Das Programm beginnt in Zelle 3004.

Adr.	Befehl	Bemerkungen
3004	3000 211	$\langle Ac \rangle := a$
3005	3001 220	$\langle Ac \rangle := a - b$
3006	3002 210	$\langle Ac \rangle := a - b + c = x$
3007	3003 240	$\langle 3003 \rangle := x$

5.3. Ausblendung der Adreßstellen

In Zelle 3000 stehe ein Befehl, dessen Adreßteil a nach Zelle 3002 gespeichert werden soll. Als Konstante sei $\langle 3001 \rangle = 0000\ 7777\ 00$ gegeben. Das Programm beginnt in Zelle 3003.

Adr.	Befehl	Bemerkungen
3003	3000 211	$\langle Ac \rangle := a$ BA Z
3004	3001 200	$\langle Ac \rangle := a^{*}$
3005	3002 240	$\langle 3002 \rangle := a^{*}$

5.4. Multiplikation mit 10

Durch ein in Zelle 3000 beginnendes Programm ist der Akkumulatorinhalt mit 10 zu multiplizieren.

Adr.	Befehl	Bemerkungen
3000	3006 644	$\langle Ac \rangle := 2^2 \cdot a = 4a$, $\langle 3006 \rangle := a$; $\langle 3023 \rangle = 2 \cdot a$
3001	3006 214	$\langle Ac \rangle := 2\,(4a + a) = 10a$

5.5. Algebraischer Ausdruck

Gegeben sei

$$\langle 3000 \rangle = a$$
$$\langle 3001 \rangle = b$$
$$\langle 3002 \rangle = c\,.$$

Es soll

$$x = 10a - 5b - 4c$$

berechnet werden. Das Programm beginne in Zelle 3003.

Für das Programm wird die Identität

$$x = 10a - 5b - 4c = 2\,[2\,(2a - b - c) + a] - b$$

benutzt.

Adr.	Befehl	Bemerkungen
3003	3000 215	$\langle Ac \rangle := 2a$
3004	3001 220	$\langle Ac \rangle := 2a - b$
3005	3002 224	$\langle Ac \rangle := 2\,(2a - b - c)$
3006	3000 214	$\langle Ac \rangle := 2\,[2\,(2a - b - c) + a]$
3007	3001 220	$\langle Ac \rangle := x$

5.6. Vordere Bits als Adresse

Gegeben sei in oktaler Darstellung

$$\langle 3000 \rangle = a_1a_2a_3a_4\ xxxx\ xxx\,.$$

Ab Zelle 3001 soll ein Programm, welches $a_1a_2a_3a_4$ als Adresse interpretiert und deren Inhalt in den Akkumulator bringt, stehen.

Adr.	Befehl	Bemerkungen
3001	3000 211	$\langle Ac \rangle := a_1a_2a_3a_4\ xxxx\ xxx$
3002	3004 642	$\langle Ac \rangle := 0000\ a_1a_2a_3a_4\ xxx$ Der von diesem W-Befehl benutzte Teil der Spur 3000 wird vom Programm nicht benötigt.
3003	0000 251	$\langle Ac \rangle := \langle a_1a_2a_3a_4 \rangle$

5.7. Trennung des Maschinenwortes in Teilwörter

In der Speicherzelle 3000 mögen nebeneinander drei Teilwörter A, B, C, die je 11 Bits lang sind, stehen:

$$\langle 3000 \rangle = (A;\ B;\ C)\,.$$

Diese Teilwörter sollen voneinander getrennt und in die vorderen 11 Bits der Speicherzellen 3001, 3002, 3003 gebracht werden, so daß

⟨3001⟩ = (A; 0; 0),
⟨3002⟩ = (B; 0; 0),
⟨3003⟩ = (C; 0; 0)

entsteht. Als Konstante stehe

⟨3004⟩ = 7776 0000 000,

als Arbeitsspur 7700 bis 7777 zur Verfügung.

Das Programm soll in Zelle 3005 beginnen.

Adr.	Inhalt	Bemerkungen
3004	7776 0000 000	Konstante
3005	3000 211	⟨Ac⟩: = (A; B; C) Programmanfang ⟵
3006	3001 240	⟨3001⟩: = (A; B; C)
3007	7721 644	11malige Linksverschiebung ⟨Ac⟩: = (B; C; 0)
3010	3002 240	⟨3002⟩: = (B; C; 0)
3011	7721 644	11malige Linksverschiebung ⟨Ac⟩: = (C; 0; 0)
3012	3003 240	⟨3003⟩: = (C; 0; 0)
3013	3004 211	⟨Ac⟩: = 7776 0000 000
3014	3001 250	⟨3001⟩: = (A; B; C) ∧ 7776 0000 000 = (A; 0; 0)
3015	3002 250	⟨3002⟩: = (B; C; 0) ∧ 7776 0000 000 = (B; 0; 0)

5.8. Betrag einer Summe

Gegeben sei ⟨3000⟩ = a, ⟨3001⟩ = b. Es soll $x = |a + b|$ gebildet und nach 3002 gespeichert werden. Das Programm beginne in Zelle 3003.

Adr.	Befehl	Bemerkungen
3003	3000 211	⟨Ac⟩: = a
3004	3001 210	⟨Ac⟩: = a + b
3005	3002 240	⟨3002⟩: = a + b
3006	3010 050	
3007	3002 221	⟨Ac⟩: = x
3010	3002 240	⟨3002⟩: = x

5.9. Ordnen eines Zahlenpaares

Gegeben sei ⟨3000⟩ = a, ⟨3001⟩ = b. Die größere der beiden positiven Zahlen soll nach Zelle 3000, die kleinere nach Zelle 3001 gebracht werden. Das Programm beginne in Zelle 3003 und ende mit einem unbedingten Stop.

Adr.	Befehl	Bemerkungen
3003	3000 211	⟨Ac⟩: = a
3004	3002 240	⟨3002⟩: = a
3005	3001 220	⟨Ac⟩: = a − b
3006	3013 050	
3007	3001 211	⟨Ac⟩: = b
3010	3000 240	⟨3000⟩: = b
3011	3002 211	⟨Ac⟩: = a
3012	3001 240	⟨3001⟩: = a
3013	0000 020	Unbedingter Stop

5.10. Addition mit Zahlenbereichsüberschreitung

Gegeben seien drei Festkommazahlen a, b, c;

⟨3000⟩ = a ≧ 0,
⟨3001⟩ = b ≧ 0,
⟨3002⟩ = c ≧ 0.

Es soll die Summe $a + b + c$ gebildet werden, falls sie den Zahlenbereich nicht verläßt, und in Zelle 3003 gespeichert werden; andernfalls soll in Zelle 3003 das Wort 4000 0000 000 gespeichert werden.

Das Programm stehe ab Zelle 3004.

Adr.	Inhalt	Bemerkungen
3004	4000 0000 000	
3005	3000 211	⟨Ac⟩: = a
3006	3001 210	⟨Ac⟩: = a + b
3007	3013 070	

3010	3002 210	⟨Ac⟩ : = a + b + c
3011	3013 070	
3012	3014 000	
3013	3004 211	⟨Ac⟩ : = 4000 0000 000
3014	3003 240	

5.11. Angabe des Vorzeichens einer Zahl

Gegeben sei ⟨3000⟩ = x. Falls x positiv ist, soll „p" gedruckt werden, falls x negativ ist „n". Das Programm beginne in 3001 und ende mit einem unbedingten Stop.

Adr.	Befehl	Bemerkungen
3001	4500 030	Anwahl der Schreibmaschine
3002	1170 030	Umschaltung auf KB
3003	3000 211	⟨Ac⟩ : = x
3004	3007 040	
3005	1270 030	Druck „p"
3006	3010 000	
3007	1050 030	Druck „n"
3010	0000 020	unbedingter Stop

5.12. Adresseneingabe

Durch Eingabe der maximal vier Oktalziffern a_1, a_2, a_3, a_4 einer Adresse soll im Akkumulator die Adresse in den Adreßstellen aufgebaut werden:

⟨Ac⟩ = 0000 $a_1a_2a_3a_4$ 000

Die Eingabe führender Nullen der Adresse soll nicht notwendig sein; da in diesem Fall weniger als vier Oktalziffern eingegeben werden, muß das Ende der Adresse durch Eingabe eines Schlußzeichens, wofür „." (Punkt) verwendet werden soll, gekennzeichnet werden. Solange die Adresseneingabe noch nicht beendet ist, soll das Zeichen „Zwischenraum" die begonnene Adresse löschen und zum Beginn der Adresseneingabe zurückführen, so daß eine Korrekturmöglichkeit vorhanden ist.

Die Eingabe erfolgte im R 300-Code.

Der Programmablauf ist durch Abb. 23 gegeben.

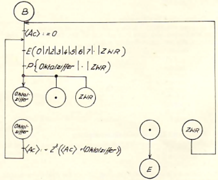


Abb. 23

Die acht verschiedenen Oktalziffern müssen in Adresseneinheiten auf einer gemeinsamen Spur, einer sogenannten Eingabespur, zur Verfügung stehen; auf derselben Spur müssen Zellen für die Zeichen „." (Punkt) und „ZWR" (Zwischenraum) vorgesehen sein.

Adr.	Inhalt	Bemerkungen
3000	3040 014	Eingabe eines Zeichens, Linksverschiebung
3001	3002 004	Linksverschiebung
3002	3000 004	Linksverschiebung
3003		Programmende (Fortsetzung nach der Adresseneingabe) ⟶
3040	0000 0000 000	Konstanten zur Eingabe der acht verschiedenen Oktalziffern
3041	0000 0000 100	
3042	0000 0000 200	
3043	0000 0000 300	
3044	0000 0000 400	
3045	0000 0000 500	
3046	0000 0000 600	
3047	0000 0000 700	
3052	3000 001	Fortsetzung nach Eingabe von „ZWR", Löschen des Akkumulators, Programmanfang ⟵
3073	3003 002	Fortsetzung nach Eingabe von „.", Übergang zum Programmende

Bei der Eingabe der Adresse 1234 wirkt das Programm in folgender Weise:

Adr.	Befehl	Bemerkungen
3052	3000 001	⟨Ac⟩ := 0
3000	3040 014	E (1), Linksverschiebung ⟨Ac⟩ := 0000 0000 200
3001	3002 004	Linksverschiebung ⟨Ac⟩ := 0000 0000 400
3002	3000 004	Linksverschiebung ⟨Ac⟩ := 0000 0001 000
3000	3040 014	E (2), Linksverschiebung ⟨Ac⟩ := 0000 0002 400
3001	3002 004	Linksverschiebung ⟨Ac⟩ := 0000 0005 000
3002	3000 004	Linksverschiebung ⟨Ac⟩ := 0000 0012 000
3000	3040 014	E (3), Linksverschiebung ⟨Ac⟩ := 0000 0024 600
3001	3002 004	Linksverschiebung ⟨Ac⟩ := 0000 0051 400
3002	3000 004	Linksverschiebung ⟨Ac⟩ := 0000 0123 000
3000	3040 014	E (4), Linksverschiebung ⟨Ac⟩ := 0000 0247 000
3001	3002 004	Linksverschiebung ⟨Ac⟩ := 0000 0516 000
3002	3000 004	Linksverschiebung ⟨Ac⟩ := 0000 1234 000
3000	3040 014	E (.) ⟨Ac⟩ := 0000 2470 000
3073	3003 002	Rechtsverschiebung ⟨Ac⟩ := 0000 1234 000
3003		Programmende

5.13. Dezimaleingabe

Es soll ein Eingabeprogramm im R 300-Code für natürliche Dezimalzahlen aufgestellt werden. Das Programm soll auf einer Spur untergebracht werden und nur eine Arbeitsspur benutzen.

Das Schlußzeichen für die einzugebende Zahl sei „." (Punkt). Durch das Korrekturzeichen „ZWR" (Zwischenraum) soll zum Anfang der Eingabe zurückgekehrt werden können (Abb. 24).

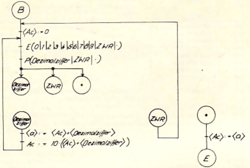


Abb 24

Adr.	Inhalt	Bemerkungen
3000	0000 0000 000	Konstanten für die einzelnen Dezimalziffern
3001	0000 0000 001	
3002	0000 0000 002	
3003	0000 0000 003	
3004	0000 0000 004	
3005	0000 0000 005	
3006	0000 0000 006	
3007	0000 0000 007	
3010	0000 0000 010	
3011	0000 0000 011	
3012	3013 001	Fortsetzung nach Eingabe vom „ZWR" ⟨Ac⟩: = 0 Programmanfang ⟵
3013	3000 010	E (0|1|2|3|4|5|6|7|8|9|ZWR|.)
3014	7706 644	zweimalige Linksverschiebung } ⟨Ac⟩: = 10 · ⟨Ac⟩
3015	7706 214	⟨7706⟩: = ⟨Ac⟩
3016	3013 000	
3033	7706 211	Fortsetzung nach Eingabe von „.", ⟨Ac⟩: = eingegebene Dezimalzahl
3034	Programmfortsetzung	

Bei Eingabe der natürlichen Dezimalzahl

12

arbeitet das Programm in folgender Weise:

Adr.	Befehl	Bemerkungen
3012	3013 001	⟨Ac⟩: = 0
3013	3000 010	E (1), ⟨Ac⟩: = 0000 0000 001
3014	7706 644	⟨Ac⟩: = 0000 0000 004 ⟨7706⟩: = 0000 0000 001

3015	7706 214	(Ac): = 0000 0000 012
3016	3013 000	
3013	3000 010	E (2), (Ac): = 0000 0000 014
3014	7706 644	(Ac): = 0000 0000 060 (7706): = 0000 0000 170
3015	7706 214	(Ac): = 0000 0000 014
3016	3013 000	
3013	3000 010	E (.)
3033	Programmfortsetzung	

5.14. Abdruck numerischer Texte vom Lochband (im Telegrafenalphabet Nr. 2)

Auf einem Lochband im Leser 2 stehe ein numerischer Text gemäß Telegrafenalphabet Nr. 2, welcher sich aus den Dezimalziffern, Vorzeichen, Komma, Zwischenraum, Wagenrücklauf und Zeilenschaltung zusammensetzt.

Dieser Text soll über das Schreibwerk abgedruckt werden. Das Schreibwerk sei auf Kleinbuchstaben geschaltet. Transportlochung soll unwirksam sein.

Das Programm benutze nur Zellen oberhalb der Adresse 2777.

Adr.	Inhalt	Bemerkungen
3000	3031 000	Programmanfang ←
3001	0230 030	Druck „3“
3002	0172 030	Zwischenvorschub
3003	3016 000	
3004	0320 030	Zwischenraum
3005	3032 000	
3006	0100 030	Druck „8“
3007	0070 030	Druck „7“
3010	1120 030	Wagenrücklauf
3012	0040 030	Druck „4“
3014	1530 030	Druck „,“
3015	3032 000	
3016	1000 030	Druck „—“
3017	3032 000	
3020	0250 030	Druck „5“
3021	0400 030	Druck „+“
3023	0020 030	Druck „2“
3025	0260 030	Druck „6“
3026	0200 030	Druck „0“
3027	0010 030	Druck „1“
3030	0310 030	Druck „9“
3031	4500 030	Anwahl des Kanals 2
3032	3000 011	Eingabe eines Zeichens
3033	3034 240	(3034): = Druckbefehl
3034		Arbeitszelle für Druckbefehl
3035	3032 000	

5.15. Abdruck numerischer Texte vom Lochband (im R 300-Code)

Auf einem Lochband im Leser 2 stehe ein numerischer Text gemäß R 300-Code, welcher sich aus den Dezimalziffern, Vorzeichen, Komma, Zwischenraum, Tabulator und Wagenrücklauf mit Zeilenschaltung zusammensetzt.

Das Schreibwerk sei auf Kleinbuchstaben geschaltet.

Das Programm benutze nur Zellen oberhab der Adresse 2777.

Adr.	Inhalt	Bemerkungen
3000	0200 030	Druck „0“
3001	0010 030	Druck „1“
3002	0020 030	Druck „2“
3003	0230 030	Druck „3“
3004	0040 030	Druck „4“
3005	0250 030	Druck „5“
3006	0260 030	Druck „6“
3007	0070 030	Druck „7“

3010	0100 030	Druck „8"
3011	0310 030	Druck „9"
3012	0320 030	Zwischenraum
3013	3015 000	
3014	4500 030	Anwahl 2 Programmanfang ⟵
3015	3000 011	Eingabe eines Zeichens (Ac) : = Druckbefehl, falls Umstellung in Addition
3016	3021 240	(3021) : = Druckbefehl
3017	3021 000	
3020	0400 030	Druck „+"
3021	Druckbefehl	
3022	3015 000	
3032	2730 030	Wagenrücklauf mit Zeilenschaltung
3033	3015 000	
3040	1000 030	Druck „–"
3041	3015 000	
3073	1530 030	Druck „ , "
3074	3015 000	
3075	1550 030	Tabulator
3076	3015 000	

Soll die erste Zelle des Programms zugleich der Programmanfang sein, so muß dort ein unbedingter Sprung zur Programmfortsetzung stehen und die Information des Druckbefehls für die Ziffer 0 zusätzlich in dieser Zelle stehen. Das ist möglich, indem die Druckbefehle für diejenigen Zeichen, welche den Eingabe-Halt-Befehl in eine Addition umstellen, um 15 Bits zyklisch linksverschoben gespeichert werden.

Adr.	Inhalt	Bemerkungen
3000	2000 3014 000	Programmanfang ⟵ Druck „0"
3001	0100 3000 000	Druck „1"
3002	0200 3000 000	Druck „2"
3003	2300 3000 000	Druck „3"
3004	0400 3000 000	Druck „4"
3005	2500 3000 000	Druck „5"
3006	2600 3000 000	Druck „6"
3007	0700 3000 000	Druck „7"
3010	1000 3000 000	Druck „8"
3011	3100 3000 000	Druck „9"
3012	0320 030	Zwischenraum
3013	3015 000	
3014	4500 030	Anwahl 2
3015	3000 017	Eingabe eines Zeichens (Ac) : = Druckkonstante, falls Umstellung in Addition mit zyklischer Linksverschiebung
3016	7720 646	16malige zyklische Linksverschiebung
3017	3021 006	(Ac) : = Druckbefehl zyklische Linksverschiebung
3020	4000 3000 000	Druck „+"
3021	3022 240	(3022) : = Druckbefehl
3022	Druckbefehl	
3023	3015 000	
3032	2720 030	Wagenrücklauf mit Zeilenschaltung
3033	3015 000	
3040	1000 030	Druck „–"
3041	3015 000	
3073	1530 030	Druck „ , "
3074	3015 000	
3075	1550 030	Tabulator
3076	3015 000	

5.16. Umspeichern eines Vektors

Die Komponenten des Vektors $\{a_1, ..., a_n\}$ seien ab Zelle a fortlaufend gespeichert. Sie sollen so umgespeichert werden, daß sie fortlaufend ab einer Zelle mit vorgegebener Adresse b stehen. Das Programm beginne in Zelle 3000. (Abb. 25)

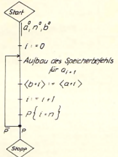


Abb. 25

Dem Programm müssen als Parameter übermittelt werden:

a° = ⟨3001⟩ Adresse der ersten Komponente des gespeicherten Vektors

n° = ⟨3002⟩ Dimension des Vektors

b° = ⟨3003⟩ Adresse der ersten Komponente des umgespeicherten Vektors

Adr.	Inhalt	Bemerkungen	
3000	3007 000	Programmanfang ⟵	
3001	a°		Parameter
3002	n°		
3003	b°		
3004	0001 000	Adresseneinheit	Konstanten
3005	0000 240	Speicherbefehl	
3006		Arbeitszelle für i	
3007	3006 241	i: = 0	
3010	3003 210	⟨Ac⟩ : = (b + i)°	Aufbau und Abspeicherung des Speicherbefehls für b_{i+1}
3011	3005 210	⟨Ac⟩ : = (b + i) 240	
3012	3016 240	⟨3016⟩ :=(b+i) 240	
3013	3006 211	⟨Ac⟩ : = i°	
3014	3001 210	⟨Ac⟩ : = (a + i)°	
3015	0000 251	⟨Ac⟩ : = a_{i+1}	⟨b + i⟩ : = ⟨a + i⟩
3016	(b+i) 240	⟨b + i⟩ : = a_{i+1}	
3017	3006 211	⟨Ac⟩ : = i°	i: = i + 1
3020	3004 210	⟨Ac⟩ : = (i + 1)°	
3021	3006 240		
3022	3002 330	Vergleicher 3, falls i = n	
3023	3025 170	Prüfen, ob i = n	
3024	3010 000		
3025	0000 020	Programmende ⟶	

5.17. Oktaldruck

Der Inhalt der Zelle a = 7777 soll in Triadenschreibweise gedruckt werden. Für das Programm stehe die Spur 3000 sowie die Arbeitsspur 7700 zur Verfügung.

Durch jeweils 3malige Linksverschiebung wird die nächste zu druckende Triade des Wortes in die drei niedrigsten Bits gebracht und ausgeblendet. Von dieser isolierten Triade wird so lange eine Einheit subtrahiert, bis eine negative Differenz entsteht. Die

erforderliche Anzahl von Subtraktionen bestimmt die zu druckende Ziffer. Ein Zähler z wird so eingestellt, daß er nach Bestimmung von 11 Oktalziffern den Wert Null hat.

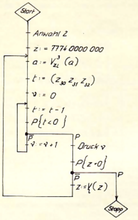


Abb 26

Adr.	Inhalt	Bemerkungen
3000	4500 030	Programmanfang ←—
		Anwahl 2
3001	3027 211	⟨Ac⟩: = 0000 0000 001
3002	3032 000	
3003	7777 217	⟨Ac⟩: = V_{zl}^2 (a)
3004	3015 006	
3005	0010 030	Druck „1“
3006	0260 030	Druck „6“
3007	7721 520	GX-Subtraktion zur Bestimmung der zu druckenden Oktalziffer
3010	7774 0000 000	Konstante
3011	3020 311	⟨Ac⟩: = z; Vergleicher 3, falls z = 0
3012	3036 170	Prüfen, ob z = 0
3013	3034 004	z: = z + 1
3014	0040 030	Druck „4“
3015	3024 006	
3016		frei
3017	0000 007	Konstante zum Ausblenden der letzten Triade
3020		Arbeitszelle für z
3021		frei
3022	0020 030	Druck „2“
3023	0070 030	Druck „7“
3024	7777 240	a: = V_{zl} (a)
3025	3017 200	⟨Ac⟩: = t
3026	3007 000	
3027	0000 001	Konstante
3030	0200 030	Druck „0“

3031	0250 030	Druck „5"
3032	7724 440	
3033	3010 211	⟨Ac⟩ : = 7774 0000 000
3034	3020 240	⟨3020⟩ : = z
3035	3003 000	
3036	3001 020	unbedingter Stop; Programmende ⟶
3037	0230 030	Druck „3"

5.18. Kleines Eingabeprogramm

Da durch den Befehl Eingabe-Halt nur die Eingabe eines einzigen Zeichens realisiert werden kann, sind für die Eingabe von Programmen spezielle Eingabeprogramme notwendig. Als besonders einfaches Eingabeprogramm ermöglicht das kleine Eingabeprogramm (KEP) die bitweise Eingabe und fortlaufende Abspeicherung von Maschinenwörtern. Es wird zur Eingabe leistungsfähigerer Eingabeprogramme benutzt. Das kleine Eingabeprogramm selbst kann nur mit technischen Hilfsmitteln eingegeben werden. Es ist auf der Bahn Null untergebracht und beginnt mit der Zelle 0002.

Alle einzugebenden Wörter setzen sich dabei aus den Zeichen „0" (duale Null) und „1" (duale Eins) zusammen; zur Markierung des Wortendes kommen dazu die Zeichen „:" (Doppelpunkt), „." (Punkt). Das Zeichen „ZWR" (Zwischenraum) bedeutet, daß eine Korrektur erfolgt. Die Wörter werden bitweise in Ac aufgebaut. Durch das Schlußzeichen „:" wird das aufgebaute Wort zum Leitbefehl erklärt. Wird ein aufgebautes Wort durch „." beendet, so wird anschließend der Leitbefehl (bei Programmeingabe ein Speicherbefehl) abgearbeitet und um eine Adresseneinheit erhöht (falls er nicht als Sprung ausgeführt wird), wonach das nächste Wort in Ac aufgebaut werden kann. Der Aufbau des Wortes beginnt beim höchsten von Null verschiedenen Bit und endet bei z_{32}. Dabei werden alle eingegebenen Bits zunächst in die Stelle z_0 eingefügt und durch sukzessive zyklische Linksverschiebung wird diese Stelle für die folgenden Bits freigemacht. Das Zeichen „ZWR" bewirkt die Löschung des Akkumulators, ohne daß der Leitbefehl verändert wird. Anschließend kann ein Wort in Ac aufgebaut werden (Abb. 27). Das KEP gestattet sowohl die Eingabe im R 300-Code als auch im Telegrafenalphabet Nr. 2.

Die Bedienung des Programmes ist für beide Codes identisch. Die Unterschiede zwischen den Codes verursachen veränderte Verarbeitung der eingegebenen Zeichen. Im folgenden wird die Arbeitsweise des kleinen Eingabeprogrammes bei Eingabe von Wörtern in Speicherzellen mit aufeinanderfolgenden Adressen beschrieben.

Adr.	Inhalt	⟨Ac⟩	Bemerkungen	
0002	0022 000		Programmanfang ⟵	
.				
.				
.				
0014	7723 211	⟨7723⟩ = ⟨7723⟩ + 1°	Erhöhung des Leitbefehls	
0015	0025 210			
0016	7723 240		⟨7723⟩ : = ⟨Ac⟩ = Leitbefehl	
0017	0030 001	⟨Ac⟩ : = 0		
0020	0000 0000 000		„0" Konstante im R 300-Code	
0021	4000 0000 000		„1" Konstante im R 300-Code	
0022	0037 211			
0023	7724 240		⟨7724⟩ : = 0014 000	
0024	0030 001	⟨Ac⟩ : = 0	„ZWR" im FS-Code	
0025	0001 000		Konstante (Adresseneinheit)	
0026	0000 0000 000		„0" Konstante im FS-Code	
0027	4000 0000 000		„1" Konstante im FS-Code	
0030	0020 010	E (0|1| : | . |ZWR)		
0031	0030 006		⟨Ac⟩ : = V^{1}_{21} (⟨Ac⟩)	
0032	0030 001	⟨Ac⟩ : = 0	„ZWR" im R 300-Code	
0033	7723 000		„." im R 300-Code	Sprung zur Ausführung des Leitbefehls
0034	7723 000		„." im FS-Code	
0035	0016 000		„:" im R 300-Code	
0036	0016 000		„:" im FS-Code	
0037	0014 000		Konstante	

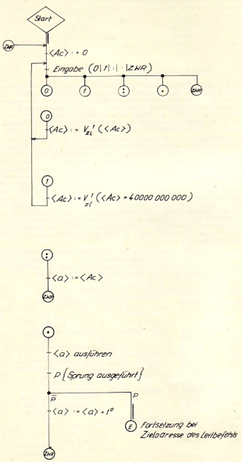

Start
ZWR
⟨Ac⟩ := 0
Eingabe (0|1|:|·|ZWR)
0
1
:
·
ZWR
⟨Ac⟩ := V'zl (⟨Ac⟩)
⟨Ac⟩ := V'zl (⟨Ac⟩ + 40000 000 000)
⟨a⟩ := ⟨Ac⟩
⟨a⟩ ausführen
p {Sprung ausgeführt}
p
p̄
⟨a⟩ := ⟨a⟩ + 1°
E
Fortsetzung bei
Zieladresse des Leitbefehls

Abb. 27

5.19. Zeitoptimale Programmierung

In Zelle 4000 stehe ein Befehl des Typs B = 2, z. B. der Lesebefehl 7716 211. Da die Sektornummer in diesem Befehl keine der beiden optimalen 32 oder 7 ist, treten bei der Abarbeitung zusätzliche Wartezeiten auf. Der zeitliche Ablauf wird durch das folgende Schema beschrieben:

Sektor-nummer	Phase	Bemerkungen
00	II	Entnahme des (Lese-)Befehls
15	III	
32		optimale Operandenadressen
07		werden nicht genutzt
24	W_2	
01		
16	IV	Ausführung der Operation
33	I	für nächsten Befehl
10		
..		
..	W_1	
..		
..		
24		
01	II	Entnahme des nächsten Befehls

Dabei wird insgesamt eine Trommelumdrehung Wartezeit verbraucht.

Falls es nicht möglich ist, die Operandenadresse optimal zu wählen, kann trotzdem durch Einbau eines unbedingten Sprunges eine Zeitverbesserung erzielt werden:

Sektor-nummer	Phase	Bemerkungen
00	II	Entnahme des Befehls xx24 000
15	III	
32	IV	
07	I	
24	II	Entnahme des (Lese-)Befehls
01	III	
16	IV	Ausführung der Operation mit zeitoptimaler Operandenadresse
33	I	für nächsten Befehl
10	W_1	
25	II	Entnahme des nächsten Befehls

Dabei werden 28 Wortzeiten eingespart.

5.19.1. Algebraischer Ausdruck

Die Aufgabenstellung sei dieselbe wie 5.5.

Das Programm soll möglichst zeitoptimal arbeiten.

Adr.	Inhalt	Bemerkungen
3003	3006 000	
3006	3000 215	$\langle Ac \rangle := 2a$
3007	3001 220	$\langle Ac \rangle := 2a - b$
3010	3002 224	$\langle Ac \rangle := 2 \cdot (2a - b - c)$
3011	3031 000	
3031	3000 214	$\langle Ac \rangle := 2 \cdot [2 \cdot (2a - b - c) + a]$
3032	3001 220	$\langle Ac \rangle := x$

In allen Rechenbefehlen werden die Operanden zeitoptimal entnommen. Während in 5.5 von der Entnahme des ersten Befehls bis zur Entnahme des letzten Operanden ca. $2^2/_3$ Trommelumdrehungen vergehen, ist es hier wegen der zeitoptimalen Programmierung genau eine Trommelumdrehung weniger.

5.19.2. Trennung des Maschinenwortes in Teilwörter

Die Aufgabenstellung sei dieselbe wie 5.7.
Das Programm soll möglichst zeitoptimal arbeiten.

Adr.	Inhalt	Bemerkungen
3004	7776 0000 000	Konstante
3005	3006 000	Programmanfang ⟵
3006	3000 211	(Ac) : = (A; B; C)
3007	3001 240	(3001) : = (A; B; C)
3010	7702 646	22malige zyklische Linksverschiebung (Ac) : = (C; A; B)
3011	3003 240	(3003) : = (C; A; B)
3012	3032 000	
3032	7702 646	22malige zyklische Linksverschiebung (Ac) : = (B; C; A)
3033	3002 240	(3002) : = (B; C; A)
3034	3035 000	
3035	3004 211	(Ac) : = 7776 0000 000
3036	3047 000	
3047	3001 250	(3001) : = (A; B; C) ∧ 7776 0000 000 = (A; 0; 0)
3050	3002 250	(3002) : = (B; C; A) ∧ 7776 0000 000 = (B; 0; 0)
3051	3003 250	(3003) : = (C; A; B) ∧ 7776 0000 000 = (C; 0; 0)

In allen Einzel-Rechenbefehlen werden die Operanden zeitoptimal entnommen. Während in 5.7 von der Entnahme des 1. Befehls bis zur Entnahme des letzten Operanden etwa 9½ Trommelumdrehungen vergehen, sind es hier wegen der zeitoptimalen Programmierung fast 5 Trommelumdrehungen weniger.

6. Wortinterpretationen

Durch entsprechende Programmsysteme können Wörter sowohl in interner als auch in externer Darstellung von der durch die Verdrahtung des Rechners unmittelbar gegebenen Interpretation abweichend interpretiert werden. Einige Beispiele für mögliche Interpretationen des Maschinenwortes werden hier angegeben.

6.1. Zahlwort „Festkomma vorn"

Das Zahlwort in der Maschine wird durch die Verdrahtung von Addition und Subtraktion als Festkommazahl definiert, wobei zunächst die Stellung des Kommas noch nicht festgelegt ist. Für die Multiplikation oder Division muß jedoch die Kommastellung definiert werden.
Steht das Komma zwischen z_0 und z_1, so ist die interne Darstellung bereits unter 3.1.1. beschrieben.

6.1.1. Multiplikation positiver Zahlen

Zu berechnen sei das Produkt a · b zweier Zahlen

$$0 \leq a < 1,$$
$$0 \leq b < 1.$$

Das Programm beginne in Zelle 3000. Die mit 7700 beginnende Spur stehe als Arbeitsspur zur Verfügung. Es sei

(3007) = a
(3010) = b.

Adr.	Inhalt	Bemerkungen
3000	3007 211	(Ac) : = a
3001	7700 440	Transport von a auf alle Zellen einer Spur
3002	3010 211	(Ac) : = b
3003	7700 472	Bildung der Teilprodukte $P_i = b_i \cdot a$ für i = 32, ..., 1
3004	3011 211	(Ac) : = 2^{-1} Rundung des Resultats
3005	7700 412	(Ac) : = a·b Summierung der Teilprodukte $\sum_{i=1}^{32} P_i \cdot 2^{-i}$
3006	3000 020	unbedingter Stop
3007	a	
3010	b	
3011	2000 0000 000	Konstante

Die von den einzelnen Teilprodukten verursachten Beiträge zum Fehlerbetrag des Produktes werden durch

$$2^{-64},\ 2^{-63},\ \ldots,\ 2^{-33}$$

beschränkt, so daß

$$2^{-64} + 2^{-63} + \ldots + 2^{-33} < 2^{-32}$$

eine Schranke für den gesamten Fehlerbetrag ist.

Zahlenbeispiel:

$$\left.\begin{array}{r} a = 0,\ 110\,101 \\ b = 0,\ 111\,001 \\ a \cdot b = 0,\ 101\,111 \end{array}\right\}\ \text{dual}$$

Dabei wird abweichend vom C 8205 angenommen, daß die Wortlänge nur 7 Bits betrage und auf der Arbeitsspur die Speicherzellen 0, 1, 2, 3, 4 und 5 in dieser Reihenfolge liegen.

Befehl	⟨0⟩	⟨1⟩	⟨2⟩	⟨3⟩	⟨4⟩	⟨5⟩	⟨Ac⟩
3000	–	–	–	–	–	–	0110101
3001	0110101	0110101	0110101	0110101	0110101	0110101	„
3002	„	„	„	„	„	„	0111001
3003	0110101	„	„	„	„	„	0011100
	„	0000000	„	„	„	„	0001110
	„	„	0000000	„	„	„	0000111
	„	„	„	0110101	„	„	0000011
	„	„	„	„	0110101	„	0000001
	„	„	„	„	„	0110101	0000000
3004	„	„	„	„	„	„	0100000
3005	*	„	„	„	„	„	0101010|1
	„	*	„	„	„	„	0010101|0
	„	„	*	„	„	„	0001010|1
	„	„	„	*	„	„	0011111|1
	„	„	„	„	*	„	0101010|0
	„	„	„	„	„	*	0101111|1

6.1.2. **Verkürzte Multiplikation positiver Zahlen**

Gegeben sei

$$\langle 3007\rangle = x = 0\ x_1\, x_2 \ldots x_{32}$$
$$\langle 3010\rangle = a$$
$$\langle 3011\rangle = 1 - 2^{-32}$$

Es soll $x \cdot a$ berechnet werden. Das Programm beginne in Zelle 3000.

Als Arbeitsspur stehe die mit 7700 beginnende zur Verfügung.

Adr.	Inhalt	Bemerkungen	
3000	3007 211	⟨Ac⟩ : = x	
3001	7700 440		Speicherung von x in alle Zellen der Arbeitsspur
3002	3011 211	⟨Ac⟩ : = 3777 7777 777	
3003	7700 462		Durch diesen Befehl werden auf der Arbeitsspur nach folgendem Schema die höchsten Bits von x als mögliche Teilprodukte ausgeblendet: $0\ 0\ 0\ 0\ \ldots 0 = \langle 7700\rangle$ $0\ x_1\ 0\ 0\ \ldots 0 = \langle 7715\rangle$ $0\ x_1\ x_2\ 0\ \ldots 0 = \langle 7732\rangle$ $0\ x_1\ x_2\ x_3 \ldots 0 = \langle 7707\rangle$ . . . . $0\ x_1\ x_2 \ldots x_{31}\ 0 = \langle 7723\rangle$
3004	3010 211	⟨Ac⟩ : = a	
3005	7700 432		Abbau des Faktors a und gleichzeitiger Aufbau des Produktes in Ac
3006	3000 020	unbedingter Stop	
3007	x		
3010	a		
3011	3777 7777 777	Maske zur Erzeugung der Teilprodukte	

Durch das angegebene Programm wird das Produkt auf 26 Dualstellen genau berechnet. Für jedes Teilprodukt wird der Fehlerbetrag durch 2^{-32} beschränkt, die Anzahl der Teilprodukte ist $32 = 2^5$, also liegt Abweichung innerhalb $2^5 \cdot 2^{-32} = 2^{-27}$. Der Fehler ist stets negativ.

Die dargestellte Durchführung der Multiplikation ist besonders vorteilhaft, wenn der Faktor x wiederholt benutzt wird; das ist z. B. beim Horner-Schema der Fall. Dann brauchen die Befehle 3000 – 3003 für mehrere Multiplikationen nur einmal ausgeführt zu werden.

Zahlenbeispiel:

$$\left.\begin{aligned} a &= 0{,}110101 \\ x &= 0{,}111001 \\ a \cdot x &= 0{,}101111 \end{aligned}\right\} \text{dual}$$

Dabei wird abweichend vom C 8205 angenommen, daß die Wortlänge nur 7 Bits beträgt und auf der Arbeitsspur die Speicherzellen 0, 1, 2, 3, 4 und 5 in dieser Reihenfolge liegen.

Die Abweichung des Resultats vom genauen Wert wird durch den Verfahrensfehler verursacht.

Befehl	⟨0⟩	⟨1⟩	⟨2⟩	⟨3⟩	⟨4⟩	⟨5⟩	⟨Ac⟩
3000	—	—	—	—	—	—	0111001
3001	0111001	0111001	0111001	0111001	0111001	0111001	"
3002	"	"	"	"	"	"	0111111
3003	0000000	"	"	"	"	"	0011111
	"	0100000	"	"	"	"	0001111
	"	"	0110000	"	"	"	0000111
	"	"	"	0111000	"	"	0000011
	"	"	"	"	0111000	"	0000001
	"	"	"	"	"	111000	0000000
3004	"	"	"	"	"	"	0110101
3005	•	"	"	"	"	"	00|11010
	"	•	"	"	"	"	000|1101
	"	"	•	"	"	"	0011|110
	"	"	"	•	"	"	00011|11
	"	"	"	"	•	"	010001|1
	"	"	"	"	"	•	01011011

6.1.3. Quadrierung einer positiven Zahl

Es soll x^2 mit $0 \leq x < 1$ berechnet werden.
Es sei

$$\langle 3006 \rangle = x\,.$$

Nach dem Verfahren von 6.1.1. ergibt sich das folgende Programm:

Adr.	Inhalt	Bemerkungen	
3000	3006 211	$\langle Ac \rangle := x$	
3001	7700 440		Transport von x auf alle Zellen einer Spur
3002	7700 472		Bildung der Teilprodukte
3003	3007 211	$\langle Ac \rangle := 2^{-1}$	Rundung des Resultats
3004	7700 412		Summierung der Teilprodukte
3005	3000 020	unbedingter Stop	
3006	x		
3007	2000 0000 000	Konstante	

6.1.4. Division positiver Zahlen

Gegeben sei

$$\langle 3010 \rangle = a \qquad \text{mit} \quad 0 \leq a < 1\,, \qquad b = 0\, b_1\, b_2 \ldots b_{32} \text{ (dual)}$$
$$\langle 3011 \rangle = b \qquad \quad 0 \leq b < 1\,,$$
$$a < b\,.$$

Es soll

$$\begin{aligned} q &= \frac{a}{b} \\ &= 0\, z_1\, z_2 \ldots z_{32} \text{ (dual)} \end{aligned}$$

berechnet werden. Das Programm beginne in Zelle 3000. Die Berechnung erfolgt nach dem Schema der ziffernweisen Bestimmung des Quotienten ohne Restrückstellung (im Dualsystem):

$$R_0 := a$$
$$R_i := 2\,R_{i-1} - b \text{ sign } R_{i-1} \qquad \text{sign } R_i = \begin{cases} +1 \text{ für } R_i \geqq 0 \\ -1 \text{ für } R_i < 0, \end{cases}$$
$$z_i := \frac{1}{2}\,(1 + \text{sign } R_i)$$
$$(i = 1, \ldots 32)$$

Ist $R_i < 0$, wird dem Ergebnis die Ziffer $z_i = 0$, ist $R_i \geqq 0$, die Ziffer $z_i = 1$ angefügt. Wegen

$$(R + b) \cdot 2 - b = 2\,R + b$$

braucht ein negativer Rest nicht durch Addition von b auf einen nicht negativen Wert zurückgestellt zu werden, um anschließend nach links zu verschieben und b zu subtrahieren, sondern es kann sofort nach links verschoben und anschließend b addiert werden.

Für das Programm wird dieser Algorithmus in einen äquivalenten umgeformt. Aus

$$r_0 = b - a > 0$$

folgt bei derselben Rekursionsformel für r_1

$$r_1 = 2\,r_0 - b = 2\,(b - a) - b = b - 2a = -R_1,$$

und wegen der Rekursionsformel

$$r_i = 2\,r_{i-1} - b \text{ sign } r_{i-1}, \quad (i = 1, \ldots, 32) \; |r_i| \leqq \frac{1}{2}\,b,$$

ist auch für $R_i \neq 0$

$$r_i = -R_i$$

und $z_i = \frac{1}{2}\,(1 - \text{sign } r_i)$

mit $\text{sign } r_i = \begin{cases} +1 \text{ für } r_i \geqq 0 \\ -1 \text{ für } r_i < 0. \end{cases}$

Adr.	Inhalt	Bemerkungen	
3000	3011 213	$\langle Ac \rangle := 2^{-1} \cdot b$	Durch die Rechtsverschiebung wird ein Verlust des Vorzeichens von r_i bei Befehl 3006 vermieden
3001	7700 440		Transport von $2^{-1} \cdot b$ auf alle Zellen einer Arbeitsspur
3002	3012 211	$\langle Ac \rangle := 7777\;7777\;777$	
3003	7700 454		Löschen der niedrigsten Bits von b in den Zellen der Arbeitsspur nach folgendem Schema: $0\;\;0\;\;b_1\;b_2 \ldots b_{30}b_{31} = \langle 7700 \rangle$ $0\;\;0\;\;b_1\;b_2 \ldots b_{30}0\;\; = \langle 7715 \rangle$ . . . . $0\;\;0\;\;b_1\;0\;\;0\;\;0\;\;\;\; = \langle 7706 \rangle$ $0\;\;0\;\;0\;\;0 \;\ldots\; 0\;\;0\;\; = \langle 7723 \rangle$
3004	3010 221	$\langle Ac \rangle := -a$	
3005	3011 210	$\langle Ac \rangle := b-a$	
3006	7700 436	$\langle Ac \rangle := \pm q$	Eigentlicher Divisionsvorgang $r_i := 2r_{i-1} - b \cdot \text{sign } r_{i-1}$ $z_i := \frac{1}{2}$ (1-sign r_i) Durch das Abschneiden der niedrigsten Bits von b ist jetzt in Ac Platz für die Ziffern z_i des Quotienten
3007	3000 020	unbedingter Stop	
3010	a		
3011	b		
3012	7777 7777 777	Maske zum Abschneiden der niedrigsten Bits von b	

Der Fehler bei diesem Näherungsverfahren ist kleiner als 2^{-26}.

Zahlenbeispiel:

$$\left.\begin{aligned} a &= 0,010000 \\ b &= 0,100000 \\ q = \tfrac{a}{b} &= 0,100000 \end{aligned}\right\} \text{dual}$$

Dabei wird abweichend vom C 8205 angenommen, daß die Wortlänge nur 7 Bits betrage und auf der Arbeitsspur die Speicherzellen 0, 1, 2, 3, 4 und 5 in dieser Reihenfolge liegen. Die Abweichung des Resultats vom genauen Wert liegt innerhalb der Fehlerschranke bei dieser Wortlänge.

Befehl	⟨0⟩	⟨1⟩	⟨2⟩	⟨3⟩	⟨4⟩	⟨5⟩	⟨Ac⟩
3000	—	—	—	—	—	—	0010000
3001	0010000	0010000	0010000	0010000	0010000	0010000	"
3002	"	"	"	"	"	"	1111111
3003	0010000	"	"	"	"	"	1111110
	"	0010000	"	"	"	"	1111100
	"	"	0010000	"	"	"	1111000
	"	"	"	0010000	"	"	1110000
	"	"	"	"	0010000	"	1100000
	"	"	"	"	"	0000000	1000000
3004	"	"	"	"	"	"	1110000
3005	"	"	"	"	"	"	0010000
3006	•	"	"	"	"	"	0000000|0
	"	•	"	"	"	"	11000|01
	"	"	•	"	"	"	1100|011
	"	"	"	•	"	"	110|0111
	"	"	"	"	•	"	11|01111
	"	"	"	"	"	•	1|011111

6.2. Ganze Festkommazahlen

Steht das Komma im Maschinenzahlwort hinter z_{32}, so wird das Wort als ganze Zahl interpretiert.

Positive Zahlen haben die Darstellung

$$z = \sum_{i=1}^{32} z_i \cdot 2^{32-i}$$

Negative Zahlen werden durch das Zweierkomplement dargestellt:

$$\tilde{z} = 2^{33} - z\,.$$

Damit sind positive Zahlen durch $z_0 = 0$, negative durch $z_0 = 1$ charakterisiert; die Null gilt als positiv. Der Zahlenbereich ist durch

$$-2^{32} \leq z \leq 2^{32} - 1$$

gegeben. Die kleinste von Null verschiedene positive Zahl ist die Eins.

6.2.1. Multiplikation

Die beiden positiven ganzen Zahlen a, b sollen miteinander multipliziert werden; es sei

$$\langle 3006\rangle = a$$
$$\langle 3007\rangle = b = 0\, b_1\, b_2 \ldots b_{32}\,.$$

Das Produkt a · b liege im Zahlenbereich.

Adr.	Inhalt	Bemerkungen
3000	3006 211	$\langle Ac\rangle := a$
3001	7700 444	Transport von a um jeweils ein Bit linksverschoben auf alle Zellen einer Arbeitsspur: $a_i := V_i^1(a)$, $(i = 0, \ldots, 31)$
3002	3007 211	$\langle Ac\rangle := b$
3003	7700 472	Bildung der Teilprodukte $P_i = b_{32-i} \cdot a_i$
3004	7700 410	$\langle Ac\rangle := a \cdot b = \sum_{i=0}^{31} P_i$
3005	3000 020	unbedingter Stop
3006	a	
3007	b	

Zahlenbeispiel:

$$\left.\begin{array}{r} b = 101 \\ a = 1\,011 \\ a \cdot b = 110\,111 \end{array}\right\} \text{ dual}$$

Dabei wird abweichend vom C 8205 angenommen, daß die Wortlänge nur 7 Bits betrage und auf der Arbeitsspur die Speicherzellen 0, 1, 2, 3, 4 und 5 in dieser Reihenfolge liegen.

Befehl	⟨0⟩	⟨1⟩	⟨2⟩	⟨3⟩	⟨4⟩	⟨5⟩	⟨Ac⟩
3000	—	—	—	—	—	—	0001011
3001	0001011	"	"	"	"	"	0010110
	"	0010110	"	"	"	"	0101100
	"	"	0101100	"	"	"	1011000
	"	"	"	1011000	"	"	0110000
	"	"	"	"	0110000	"	1100000
	"	"	"	"	"	1100000	1000000
3002	"	"	"	"	"	"	0000101
3003	"	"	"	"	"	"	0000010
	"	0000000	"	"	"	"	0000001
	"	"	"	"	"	"	0000000
	"	"	"	0000000	"	"	0000000
	"	"	"	"	0000000	"	0000000
	"	"	"	"	"	0000000	0000000
3004	•	"	"	"	"	"	0001011
	"	•	"	"	"	"	0001011
	"	"	•	"	"	"	0110111
	"	"	"	•	"	"	0110111
	"	"	"	"	•	"	0110111
	"	"	"	"	"	•	0110111

6.3. Gleitkommazahlen

Gleitkommazahlen basieren auf der Darstellung

$$z = m \cdot B^e$$

$$B^{-1} \leq |m| < 1 \quad \text{für } z \neq 0$$

wobei die Mantisse m die Ziffernfolge von z angibt, der Exponent e die Stellenwertigkeit der Mantisse festlegt und B die Basis des verwendeten Zahlensystems ist. Grundlage für die interne Darstellung ist die Schreibweise

$$z = m \cdot 2^e$$

mit der normalisierten Mantisse

$$0{,}5 \leq |m| < 1 \quad \text{für } z \neq 0$$

Das Maschinenwort wird in zwei Bitgruppen, die der Mantisse und dem Exponenten entsprechen, interpretiert. In den Bits z_0, z_7, z_8, ... z_{32} wird die vorzeichenbehaftete Mantisse

$$m = \pm m_1\, m_2 \dots m_{26}$$

dargestellt; ihr Vorzeichen steht in z_0, ihr Betrag $m_1\, m_2 \dots m_{26}$ in z_7, z_8, ... z_{32}. Der Exponent wird intern um $32 = 2^5$ erhöht als Maschinenexponent

$$e' = e + 32 = \sum_{i=0}^{5} e_i\, 2^i$$

in den Bits z_1, z_2, ... z_6 dargestellt. (Abb. 28)

Die Darstellung der Null ist eindeutig in der Form

$$m_i = 0 \quad \text{für } i = 1, \dots, 26$$

$$e = -32, \text{ d. h. } e' = 0\,.$$

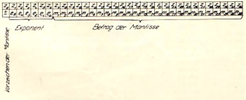


Abb. 28

Damit ist die interne Darstellung der Null bei allen angegebenen Wortinterpretationen dieselbe.

Der Bereich der zulässigen Exponenten ist durch

$$0 \leqq e' \leqq 63\,,$$

also $\quad -32 \leqq e \leqq 31$

bestimmt, der Mantissenbereich ist für $z \neq 0$ durch

$$0{,}5 \leqq |m| \leqq 1 - 2^{-26}$$

beschränkt. Folglich ist der interne Zahlenbereich durch

$$2^{-33} = 0{,}5 \cdot 2^{-32} \leqq |z| \leqq (1 - 2^{-26}) \cdot 2^{31}$$

und $\quad z = 0$ gegeben.

Beispiel:

Die interne Darstellung von

$$z = -16 = -0{,}5 \cdot 2^5$$

lautet $\quad$ 1 $\quad$ 100101 $\quad$ 100 000 000 000 000 000 000 000 00

Vorzeichen $\quad$ Exponent $\quad$ Mantissen-betrag

6.3.1. Betragsbildung

Der Inhalt x der Zelle 7777 werde als Gleitkommazahl interpretiert. Im Akkumulator ist |x| zu bilden.

Adr.	Inhalt	Bemerkungen
3000	7777 215	$\langle Ac \rangle := V_1^1$ (x), Linksverschiebung
3001	3002 002	$\langle Ac \rangle := |x|$, Rechtsverschiebung
3002	3000 020	unbedingter Stop, Programmende

6.3.2. Vorzeichenumkehr

Das Vorzeichen einer in Zelle 7777 stehenden Gleitkommazahl x ist im Akkumulator umzukehren. Das Programm soll in Zelle 3000 beginnen.

Adr.	Inhalt	Bemerkungen
3000	7777 211	$\langle Ac \rangle := x$
3001	7777 340	Vergleicher 3, falls $x = 0$
3002	3005 170	Prüfen, ob $x = 0$
3003	3006 210	$\langle Ac \rangle := x + 4000\,0000\,000 = -x$
		$z_0 = \begin{matrix} 0 \text{ für } x < 0 \\ 1 \text{ für } x > 0 \end{matrix}$ Überlauf
3004	3005 070	
3005	3000 020	unbedingter Stop, Programmende ⟶
3006	4000 0000 000	Konstante

6.3.3. Ordnen eines Zahlenpaares nach dem Betrag

Gegeben seien zwei Gleitkommazahlen

$$a = \langle 3000 \rangle$$
$$b = \langle 3001 \rangle$$

Diejenige von ihnen, welche Max. (|a|, |b|) realisiert, soll nach 3000, die andere nach 3001 gebracht werden.

Adr.	Inhalt	Bemerkungen
3002	3000 215	$\langle Ac \rangle := V_1^1$ (a)
3003	3006 642	$\langle 3023 \rangle := |a|$
3004	3001 215	$\langle Ac \rangle := V_1^1$ (b)
3005	3007 002	$\langle Ac \rangle := |b|$
3006		Arbeitszelle
3007	3023 330	Vergleicher 1, falls $|b| \leqq |a|$
3010	3017 140	Prüfen, ob $|b| \leqq |a|$
3011	3000 211	
3012	3006 240	$\langle 3006 \rangle := a$
3013	3001 211	
3014	3000 240	$\langle 3000 \rangle := b$
3015	3006 211	
3016	3001 240	$\langle 3001 \rangle := a$
3017	3002 020	unbedingter Stop, Programmende ⟶

Anhang 1: Telegrafenalphabet Nr. 2

Lochkanal 5	4		3	2	1	Eingabe-Adresse	Buchstaben	Ziffern u. Zeichen	Stanz-Adresse
●	●	·				03 *	A	–	30 x
●		·		●	●	31 *	B	?	23 x
	●	·	●	●		16	C	:	16 x
●		·		●		11	D		22 x
●		·				01	E	3	20 x
●		·	●	●		15	F		26 x
	●	·		●	●	32 *	G		13 x
		·	●		●	24	H		05 x
	●	·	●			06	I	8	14 x
●	●	·		●		13	J		32 x
●	●	·	●	●		17	K	(	36 x
	●	·			●	22 *	L	)	11 x
		·	●	●	●	34 *	M	.	07 x
		·	●	●		14	N	,	06 x
		·		●	●	30	O	9	03 x
	●	·	●		●	26	P	0	15 x
●	●	·	●		●	27	Q	1	35 x
	●	·		●		12	R	4	12 x
●		·	●			05	S	'	24 x
		·			●	20	T	5	01 x
●	●	·	●			07	U	7	34 x
	●	·	●	●	●	36	V	=	17 x
●	●	·			●	23	W	2	31 x
●		·	●	●	●	35	X	/	27 x
●		·	●		●	25	Y	6	25 x
●		·			●	21	Z	+	21 x
		·		●		10	Wagenrücklauf		02 x
	●	·				02	Zeilenvorschub		10 x
●	●	·	●	●	●	37	Buchstaben		37 x
●	●	·		●	●	33 *	Ziffern u. Zeichen		33 x
		·	●			04	Zwischenraum		04 x
		·				00	Transportlochung		00 x

* bedeutet Umstellung in Sprung

Anhang 2: R 300-Lochbandcode

Lochkanal 8	7	6	5	4		3	2	1	Eingabe Adresse	Zeichen KB	Zeichen GB	Ausgabe-Adresse
			●		·				0000	0	#	020 ×
					·			●	0001	1	<	001 ×
					·		●		0002	2	"	002 ×
			●		·		●	●	0003	3	≈	023 ×
					·	●			0004	4	[	004 ×
			●		·	●		●	0005	5	>	025 ×
			●		·	●	●		0006	6	'	026 ×
					·	●	●	●	0007	7	≋	007 ×
				●	·				0010	8	~	010 ×
			●	●	·			●	0011*	9	]	031 ×
			●	●	·		●		0012*	¤	ZWR	032 ×
				●	·	●		●	0015	:	C	015 ×
		●			·				0020	+	=	040 ×
		●	●		·			●	0021	a	A	061 ×
		●	●		·		●		0022	b	B	062 ×
		●			·		●	●	0023	c	C	043 ×
		●	●		·	●			0024	d	D	064 ×
		●			·	●		●	0025	e	E	045 ×
		●			·	●	●		0026	f	F	046 ×
		●	●		·	●	●	●	0027	g	G	067 ×
		●	●	●	·				0030	h	H	070 ×
		●		●	·			●	0031*	i	I	051 ×
		●	●	●	·		●	●	0033*	.	:	073 ×
	●				·				0040	—	△	100 ×
	●		●		·			●	0041	j	J	121 ×
	●		●		·		●		0042	k	K	122 ×
	●				·		●	●	0043	l	L	103 ×
	●		●		·	●			0044	m	M	124 ×
	●				·	●		●	0045	n	N	105 ×
	●				·	●	●		0046	o	O	106 ×
	●		●		·	●	●	●	0047	p	P	127 ×
	●		●	●	·				0050	q	Q	130 ×
	●			●	·			●	0051*	r	R	111 ×
	●	●	●		·				0060	,	;	160 ×
	●	●			·			●	0061	/	)	141 ×
	●	●	●	●	·	●			0074*	%	*	174 ×
	●	●		●	·		●	●	0073	)	?	153 ×
	●	●			·		●		0062	s	S	142 ×
	●	●	●		·		●	●	0063	t	T	163 ×
	●	●			·	●			0064	u	U	144 ×
	●	●	●		·	●		●	0065	v	V	165 ×
	●	●	●		·	●	●		0066	w	W	166 ×
	●	●			·	●	●	●	0067	x	X	147 ×
	●	●		●	·				0070	y	Y	150 ×
	●	●	●	●	·			●	0071*	z	Z	171 ×
●		●	●	●	·		●		0032*	WR2V/Satzmarke		272 ×
		●		●	·	●	●	●	0037*	GB		057 ×
	●			●	·	●	●	●	0057*	KB		117 ×
	●	●		●	·	●		●	0075*	Tab / Wortmarke		155 ×
				●	·	●	●		0016	Irrung Satz		016 ×
●	●	●	●	●	·	●	●	●	xxxx**	Irrung Zeichen		377 ×
	●	●	●	●	·	●	●	●	0077	Transport		177 ×
●	●	●		●	·		●		0072*	Blockmarke		352 ×
		●	●	●	·	●	●		0036*	Irrung Block		076 ×
	●			●	·		●		0052	W R		112 ×
	●		●	●	·	●	●		0056	Umsch. I-II Pr.		136 ×
			●	●	·	●	●	●	0017	Umsch. II-I Pr.		037 ×
	●	●	●	●	·		●		0072	Z V		172 ×
					·					Transportloch im Vorspann		000 ×

* bedeutet Umstellung in Sprung
** wird nicht eingegeben

Berichtigungen zu Heft 2

Seite 3:

Anhang 1: Telegrafenalphabet Nr. 2

Anhang 2: R 300-Lochbandcode

Seite 18, Punkt 4.3.1.7.

$\langle a \rangle := \langle Ac \rangle \wedge \langle a \rangle$

Punkt 4.3.1.8.:

$\langle a \rangle := \overline{\langle Ac \rangle} \wedge \langle a \rangle$

Seite 19, Punkt 4.3.1.10., letzte Zeile:

... nur 44 verschiedene Wirkungen.

Seite 22, 3. Zeile:

$(1 \cdot 2 \cdot 4 \cdot 6) \vee (\bar{1} \cdot 2 \cdot 4 \cdot \bar{7}) \vee (1 \cdot 2 \cdot 4 \cdot 7) \vee (\bar{1} \cdot \bar{2} \cdot \bar{3} \cdot \bar{4} \cdot 7) \vee (1 \cdot \bar{2} \cdot 3 \cdot 4) \vee (2 \cdot 4 \cdot 6 \cdot 8)$

Abb. 18:

✓ 2^4 2^3 2^2 2^1 2^0

Seite 33, Nr. 110:

... ohne Löschen, ohne Resultatbehandlung

Seite 45, Punkt 5.9., 3. Zeile:

✓ ... Zelle 3003 ...

Seite 49, 6. Zeile:

✓ ... $\langle Ac \rangle :=$ 0000 0000 170

Seite 51, letzte Zeile:

✓ ... wird so lange eine Einheit ...

Seite 53, 11. Zeile von unten:

0027 4000 0000 000 ...

Seite 55, 6. Zeile von unten:

... $\langle Ac \rangle := 2 \cdot [2 \cdot (2a - b - c) + a]$

Seite 58, Punkt 6.1.3., 4. Zeile von unten:

3004 7700 412 ...

Seite 59, Programm, 1. Zeile:

3000 3011 213 ...

Seite 60, Punkt 6.2., 4. Zeile:

$$z = \sum_{i=1}^{32} z_i \cdot 2^{32-i}$$

Seite 61, 4. Zeile:

$a \cdot b = 110\,111$

Seite 62, 1. Zeile:

... angegebenen Wortinterpretationen

Seite 62, Punkt 6.3.2., 4. Zeile von unten:

1 für $x > 0$

Seite 63, Überschrift:

Anhang 1: Telegrafenalphabet Nr. 2

Seite 64, Überschrift:

Anhang 2: R 300-Lochbandcode

dri 210-E-V2-11-Ag25-167-69

VEB KOMBINAT ZENTRONIK
Rechenelektronik Meiningen, Sitz Zella-Mehlis
DDR - 606 Zella-Mehlis
Deutsche Demokratische Republik

Exporteur:
BUROMASCHINEN-EXPORT GmbH.
DDR - 108 Berlin 8
Friedrichstraße 61